Quick Start
for

WinDraft

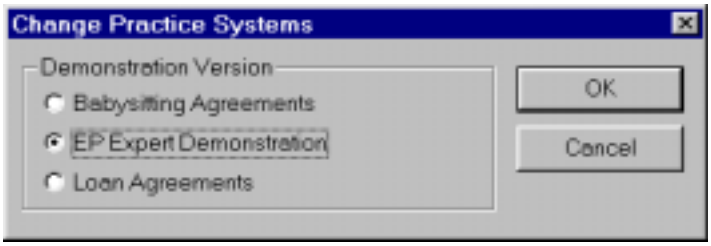
These instructions will help you to quickly learn the WinDraft basics and start producing client documents with WinDraft.

(If you have not already installed WinDraft, see the Addendum at the end of this chapter. Also, please note that this document does not walk you through creating your own system. For that, see the Author's Guide.)

1. Load Microsoft Word.

Click on the LoadWinDraft button or simultaneously press **Control-Alt-W** to "load" the Windraft global add-ins. You will see the WinDraft toolbars appear. (If you don't see the Load WinDraft button, you can load it by clicking the right mouse button while the cursor is on any toolbar.)

2. If you have access to more than one practice system in your office, select the system you want to use by clicking on the "change systems" button. The selection screen will look like this:



Your systems will vary, depending on what you or your staff have set up on your network or PC. Select the system you want to use. If you are new to WinDraft, choose EPExpert. (If you only use one system, you can skip this step. The last system you used will "stick.") If you have just installed WinDraft and are exploring

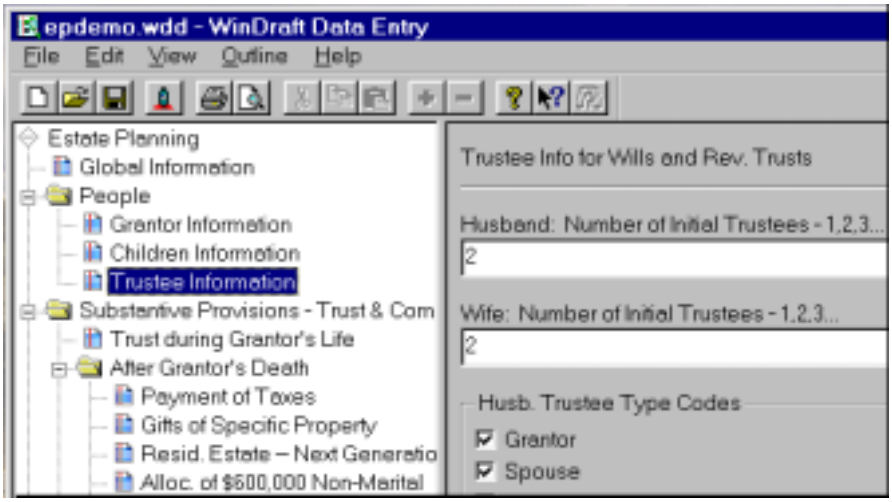
WinDraft for the first time, we recommend going through these steps with each of the demo systems. These steps will be the same as those outlined here, but the screens will look a little different for each practice system.

3. Click on the "Checklist" Button on the toolbar.

   (If it is not visible, see the Addendum at the end of this chapter or try pressing Control+Alt+W again.)

4. Use the WinDraft checklist program to enter, edit, save and print client information by following these steps:

   a. Use File Open or click on the toolbar button to open a data file. Files with a ".WDD" extension are WinDraft data files, and files with a ".WDM" extension are WinDraft model data files. (If you are using DOCS OPEN, the interface will be the usual DOCS profiles, with special document types for different practice systems.) You may use either. By loading a previous client's data, you can save time by not having to reenter data that is the same for more than one client.



b. Edit the data to suit your needs, making sure to answer all the relevant questions. There is an outline-based interface. Simply click on the folders and pages on the left side of the screen to open up data entry screens on the right side of the page. Use the tab keys or the mouse to move around the data entry screen.

You will note that questions that don't apply may be "grayed out." That is, if you select "None" for Marital Deduction Formula type, all further questions relating to the Marital Deduction and marital trusts will be displayed in gray on the screen. You cannot move the insertion point to a "grayed out" field.

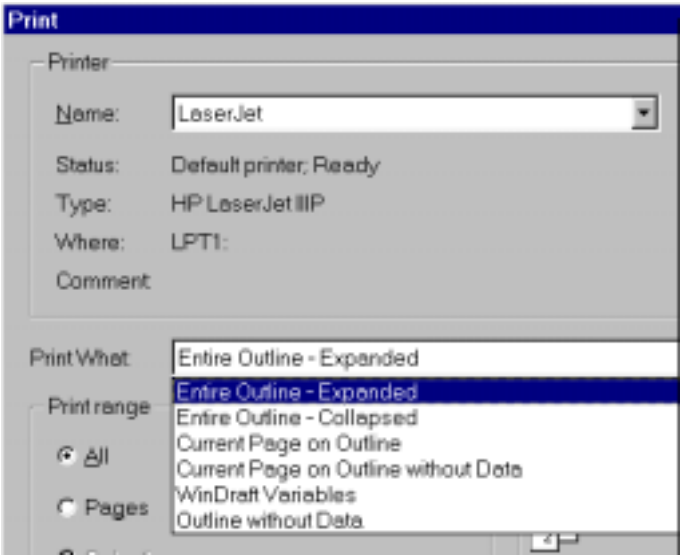
Unanswered questions are indicated as follows:

Text boxes	Contain the word, "UNANSWERED"
Radio buttons	None selected
Check boxes	Shaded gray check

Be sure to answer all the relevant questions in each section of the outline. If you leave a needed question unanswered you will receive a warning message when you merge the documents, and you will have to go back and answer it then. There are three ways to move between data entry questions. Simply clicking the mouse on the outline headings on the left side of the screen will navigate you around the various sections. The Tab key, also, can be used to move you to the next question on the screen, and Shift Tab will move you back. Outline Find Next Unanswered (Ctrl+F) is also very helpful, as pressing it takes you to the next relevant unanswered question on the data entry screen. And you can alternate between the outline and data entry side by selecting Jump to other side or (Ctrl+J).

c Save your data. File Save As will save your modified data file, using any name you choose, with up to eight characters, followed by .WDD for client data or .WDM for model data. We suggest using the client's last name as the file name so that the appropriate file can easily be recognized. If you want to save under the same name, use File Save or click on the Save button.

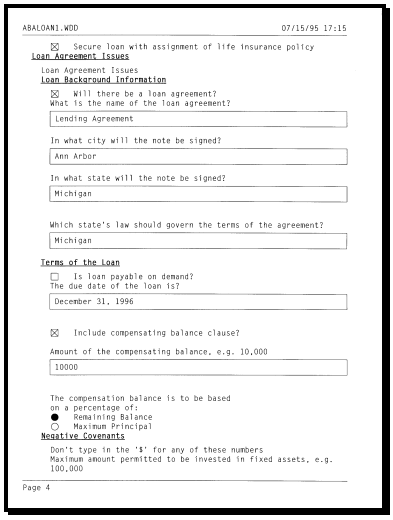
d. You can print your data either by clicking on the Print button or by selecting File Print from the menu.. If you want to review the print output on the screen, you can select File Print Preview. Note that there are six printing selections to choose from:



Normally, print the expanded outline to see your data presented in a "lawyer-friendly" way. **We recommend putting a copy of the data outline printout in the file** so that, as the documents are proofread, a lawyer can verify the selected options. Standard prompts and data will appear in black, indented to the right outline level. Formatting of the font indicates special information as follows:

Type of Information	Font
Headings	**Bold and Underlined**
Inapplicable/Irrelevant	Light gray
Unanswered	*Italic*

A typical printout with data included looks like this:



ABALOAN1.WDD 07/15/95 17:15

☒ Secure loan with assignment of life insurance policy

Loan Agreement Issues

Loan Agreement Issues

Loan Background Information

☒ Will there be a loan agreement?
What is the name of the loan agreement?

Lending Agreement

In what city will the note be signed?

Ann Arbor

In what state will the note be signed?

Michigan

Which state's law should govern the terms of the agreement?

Michigan

Terms of the Loan

☐ Is loan payable on demand?
The due date of the loan is?

December 31, 1996

☒ Include compensating balance clause?

Amount of the compensating balance, e.g. 10,000

10000

The compensation balance is to be based on a percentage of:
● Remaining Balance
○ Maximum Principal

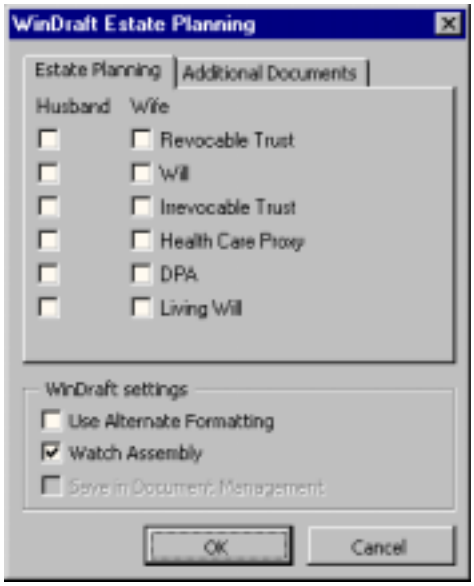
Negative Covenants

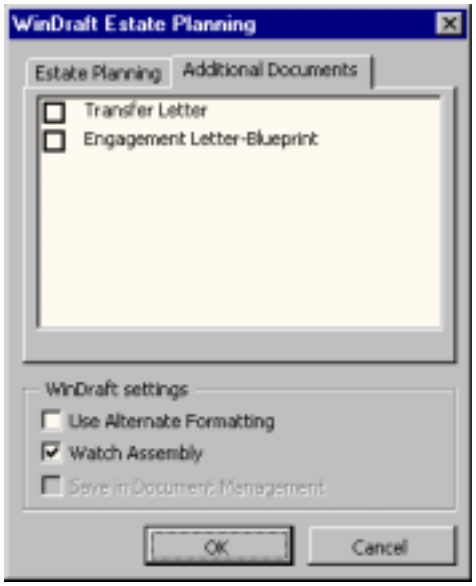
Don't type in the '$' for any of these numbers
Maximum amount permitted to be invested in fixed assets, e.g. 100,000

Page 4

To print out a blank outline for use by lawyers who are not on the system, print Outline without Data, or better yet, a partially completed outline with typical data selected. This feature lets computer-phobic attorneys use the system in a paper format. For debugging systems, you can set "Print What" to "variables" on the print dialog box (See above) in order to print the names of all the variables and their values.

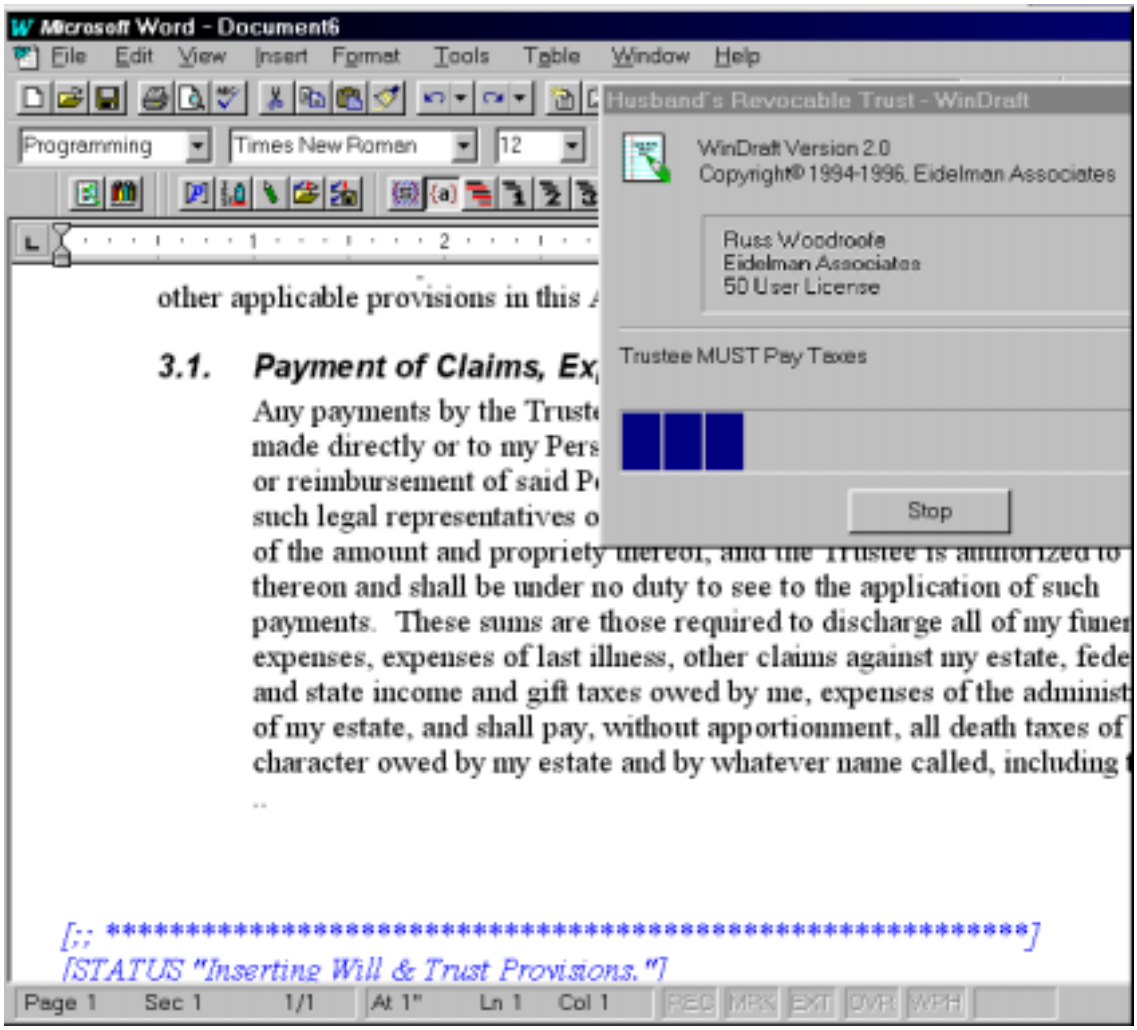
6. ***BLAST OFF!*** Click on the Rocket Icon to switch to Microsoft Word and access the "Blast Off" Dialog Box. The estate planning system looks like this:





Select the documents you want WinDraft to create, and click OK. Although it will take slightly longer, we recommend that you "Watch Assembly" so that you can observe the document as it is assembled on the screen.

The system will then merge the model documents and the client data you entered to create, on the screen, the selected documents. During the merging process it displays a progress dialog box that shows the percent (%) completed and status information, like this:


Microsoft Word - Document6
File Edit View Insert Format Tools Table Window Help
Programming
Times New Roman
12
Husband's Revocable Trust - WinDraft
WinDraft Version 2.0
Copyright© 1994-1996, Eidelman Associates
Russ Woodroofe
Eidelman Associates
50 User License
Trustee MUST Pay Taxes
Stop
other applicable provisions in this
3.1. Payment of Claims, Ex
Any payments by the Truste
made directly or to my Pers
or reimbursement of said P
such legal representatives o
thereon and shall be under no duty to see to the application of such
payments. These sums are those required to discharge all of my funer
expenses, expenses of last illness, other claims against my estate, fede
and state income and gift taxes owed by me, expenses of the administ
of my estate, and shall pay, without apportionment, all death taxes of
character owed by my estate and by whatever name called, including
[;; **********]
[STATUS "Inserting Will & Trust Provisions."]
Page 1 Sec 1 1/1 At 1" Ln 1 Col 1

When it is all done, you will see a "Document Assembly Completed!" message, and you can save the documents just like any other Word document. You have now created your personalized document in a fraction of the time it used to take and with fewer errors. (If you are running DOCS OPEN and have DOCS OPEN profile creation turned on and set up properly, the

system will create a profile for each document and display a pick list of the documents that WinDraft just created. See Appendix 4, Integration With DOCS OPEN Document Management.)

7. Editing and saving your documents

WinDraft integrates with Word very well and supports the use of MS Word bookmarks and fields, automatic paragraph numbering, table of contents, cross referencing, and outline headings. In fact, many of our model documents use these features extensively. WinDraft Edit Toolbar (an additional editing toolbar) has special styles, macros, and buttons to make it easier to work with these features.

For example, clicking on the show field shading ( ) button will turn on and off the shading of Word fields that change terminology in a document (he/she, trust/will, etc.), and the table of contents button ( ) will automatically insert a table of contents at the end of the document. The numbers on the toolbar automatically indent text in an outline format. See the WinDraft User Guide for more information. Select View | WinDraft Options from the WinDraft Checklist program and change the appropriate checkbox to load or unload the toolbar.

Addendum

1. Installing WinDraft

   Insert the WinDraft diskette in drive a:. Exit from WinWord. From Windows Program Manager or File Manager, select File, Run, "A:SETUP," and click on OK. For a standard single-user setup, we suggest going with the default. That is, put WinDraft on C:\WINDRAFT directory. For network setups, make sure you are logged in with a supervisor's ID, and read Appendix 2, Network Setup Notes, and Appendix 3, WinDraft.INI Settings.

   For further information about the WinDraft Setup program, see Appendix 2, Installing WinDraft.

2. Performance Issues

   An optimized computer operates 20 times faster than a slow, non-optimized one. Processor speed, video speed, video drivers, settings in WinWord6.ini, and other factors can each double the speed of WinDraft. See Appendix 1, Optimizing WinDraft Performance.

3. Troubleshooting

   If you have any problems, please call Eidelman Associates at 734-769-1500.

   - Different versions of WinDraft are available for Word 6.0, Word 7.0 (Office 95), Word 8.0 (Office 97), and Word 9.0 (Office 2000).
   - If you are running the system for the first time, don't forget to open the "Load WinDraft" icon or press Control-Alt-W to load and unload WinDraft. If you don't see the toolbar, you may need to open the WinDraft tool bar using the dialog under the Tools|WinDraft Options menu.
   -

WinDraft™

Expert Document Drafting Software
For Microsoft Word

User's Guide

Eidelman Associates

Microsoft Certified Solution Provider

317 South Division
Suite 187
Ann Arbor, Michigan 48104

734-769-1500 (Voice)
734-769-1501 (Fax)
support@lawtech.com
www.lawtech.com

WinDraft program, macros, field programming and Documentation Copyright © 1994-2000 Eidelman Associates, all rights reserved.

No part of this publication may be stored in retrieval systems, transmitted or reproduced in any way, including but not limited to photocopy, photograph, magnetic or other record, without prior agreement and written permission of Eidelman Associates.

WinDraft and EP Expert are trademarks of Eidelman Associates. All other brand and product names are trademarks or registered trademarks of their respective companies.

Printed 4/5/01

WinDraft
User's Guide
Table of Contents

I. THE WINDRAFT DOCUMENT ASSEMBLY SYSTEM 1

A. Scope 1

B. What is WinDraft? 1

C. The Concepts Behind WinDraft 2

II. INTRODUCTION TO WINDRAFT 3

A. WinDraft Interface 3

B. Using WinDraft Data Entry: An Overview 3

C. Terminology 3
1. Practice System 4
2. Model Document 4
3. Data Entry Screen -- The Checklist 4
4. Data File 4
5. Merged Document 4

III. USING WINDRAFT 5

A. Load Microsoft Word 5

B. Load WinDraft 5

C. Select Practice Systems 5
1. Practice Systems Button 5
2. "Checklist" Button 6
3. Exploring the Outline 6

IV. ENTERING CLIENT INFORMATION 8

A. File Names 8

B. Answering Questions 9
1. Text Boxes – Normal, Long, and Masked 10
2. Check Boxes 11
3. Choices – Radio Buttons and Dropdowns 11
4. Dates 12
5. Buttons 12

C. Advanced Navigation 13
1. Searching for Questions 13

2. Bookmarks 13

V. SAVING CLIENT FILES 14

A. Saving WinDraft Files 14

VI. BLAST OFF TO MERGE NEW DOCUMENTS 15

A. Merge (Blast Off!) 15
1. Select Document Format 15
2. Assembly 16
3. Working with Automatic Document Summaries 17

VII. PRINTING 20

A. Printing WinDraft Data 20
1. Outline -- Expanded 20
2. Outline -- Collapsed 21
3. Current Page 21
4. Variables 21
5. Outline Without Data 21

B. Printing the Final Document 21

VIII. SHARING INFORMATION WITH OTHER USERS 21

A. Emailing WinDraft Data 21

B. Emailing the Final Document 21

IX. FORMATTING IN WORD 23

A. Styles and Templates -- The Concept 23

B. The Edit Tools Toolbar 23
1. "Field" Variables 23
2. Styles for Formatting And Outline Headings 24
3. Formatting Body Text 25
4. Table of Contents 26

X. WINDRAFT OPTIONS 27

A. WinDraft Toolbar Settings 27
1. WinDraft Author toolbar 27
2. WinDraft Editing toolbar 27

B. WinDraft Default Settings 27
1. Watch document assembly 27
2. Watch hidden parts of document assembly 28

XI. HOW TO READ MODEL DOCUMENTS 28

A. WinDraft Syntax and Concepts 28
1. Inserting or "Merging" a Variable 28

2. Conditional Text or “Rules” 28
3. Document Structure 29
4. Estate Planning Documents 29

B. WinDraft Model Document Command Summary 30

WinDraft User's Guide

I. The WinDraft Document Assembly System

A. Scope

This manual concisely explains the basic techniques for using the WinDraft system. It assumes that the user is generally familiar with using Microsoft Windows and Microsoft Word. Except where noted, WinDraft uses standard Windows menu choices, tool bar buttons, check boxes, radio buttons, and other typical Microsoft Windows functions and commands in their standard ways.

If you are not familiar with Word and Windows, you may need to consult the Microsoft manuals for these programs, or some of the excellent third party guides that have been written for them, as you read this chapter.

The WinDraft User's guide assumes you already have WinDraft installed. If you have not yet installed WinDraft on your system yet, go to Appendix 2, Installing WinDraft, and follow the instructions.

B. What is WinDraft?

The WinDraft Document Assembly System is a "smart form book" that "thinks like a lawyer" as it creates your customized legal documents. It can assemble any kind of document in a simpler and far more powerful manner than a word processing program. It can be used to create an expert system, capturing much of the firm's legal knowledge and experience and making it available to everyone.

WinDraft consists of two parts: (1) data entry checklists and (2) model documents. It operates by asking the user a customized series of legal and client questions in a data entry outline. Then, it uses the answers to organize and merge a complete legal document or set of documents. The program continuously adjusts itself as information is entered. In the end, the information entered is merged with the model document so that client information is merged with the relevant legal clauses.

The system is designed to accept input data in a user-friendly way, while using a modular, easily modifiable template to design the data entry checklist outline. This feature enables attorneys to further customize their systems and update the program, if necessary.

Furthermore, WinDraft functions as an add-in to Microsoft Word for Windows. Any document that you work with or create is a Word document and can be edited as such. WinDraft also seamlessly integrates with DOCS Open, Soft Solutions, iManage, and other document management software. Finally, WinDraft can also keep logs of its work and automatically create billing and calendar entries.

C. The Concepts Behind WinDraft

WinDraft allows lawyers, secretaries and paralegals to have powerful tools at their fingertips to draft standard documents of all types. This is done through a combination of:

- Advanced features of Microsoft Word, and
- WinDraft, our document assembly engine for Word.

WinDraft is designed to feel like an add-in to Word, rather than a separate program, so anyone that understands Word also understands WinDraft.

The concept behind WinDraft is to place the expertise and accumulated knowledge of a firm at the hands of all attorneys and staff to maximize their efficiency when drafting documents. WinDraft codifies the attorney's decision-making logic so that even a non-expert can produce the very same work product. The system allows a lawyer to leverage his or her knowledge, as well as time.

To make WinDraft lawyer-friendly, the model documents resemble legal documents rather than computer programs. All complex functions can be hidden in subroutines, and the model document is basically a legal form with optional items marked and encased in brackets, with easy-to-understand notes as to what should be inserted and when. Effectively employing WinDraft requires legal knowledge, not computer knowledge.

WinDraft was designed to address the most important issues and problems facing document assembly programs. A system like this must be both *auditable* and *flexible*. It must be auditable in the sense that a lawyer can read it and understand its compliance with the law and proper procedural and substantive requirements. It must be flexible in the sense that when changes occur in the law, in the style of the attorneys, in the firm's personnel or in the needs of clients, the system can be easily updated. If it is not flexible, lawyers will start squirreling away their favorite forms again, and the system will become brittle and fall into disuse.

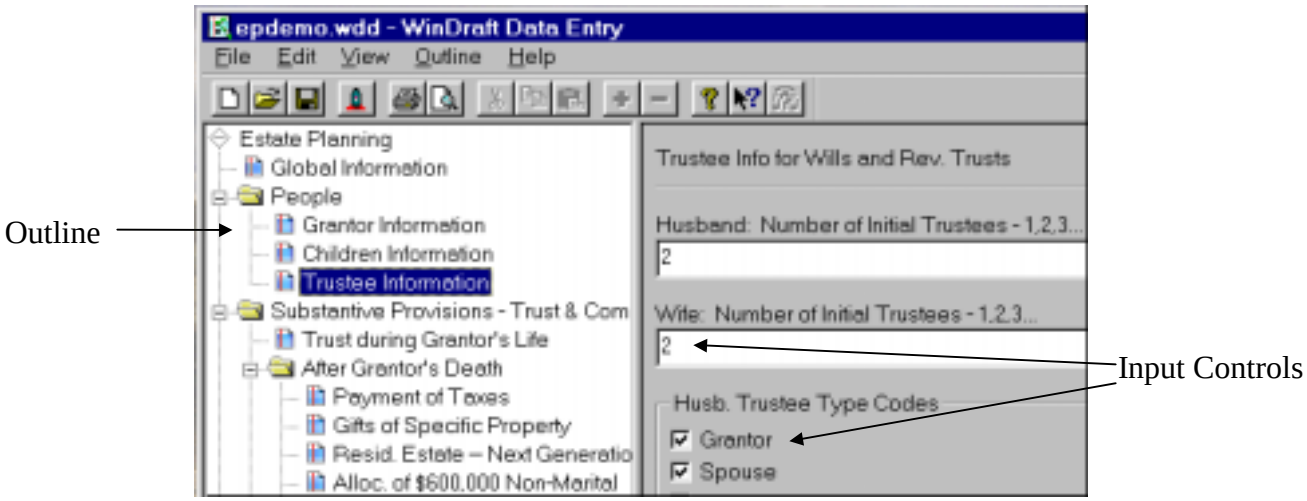
By keeping the entire system in pure Microsoft Word, any lawyer or staff member who can use Word can be a system author. It also means the firm can use all of the features of Word. The concept is a document assembly system that is designed to be as self-supporting as possible. Most of the documents look like legal documents, and the "programming" of "rules" looks, to a lawyer, as much as possible like the legal documents and rules that lawyers use every day. The goal is to let the lawyer focus on the lawyering aspects.

Lawyers live by the Rules of Ethics, rules of procedure and rules of the firm. They are centrally involved with rules of all kinds in their professional activities. WinDraft is simply a way of inserting legal rules that affect document drafting, formatting rules and other rules into the word processing system.

II. Introduction to WinDraft

A. *WinDraft Interface*

WinDraft operates by presenting the user with a series of data entry screens that look like this:



The user navigates through an outline on the left side of the screen and answers questions on the right side of the screen, using typical Windows controls (text boxes, check boxes, and radio buttons.) As you move down the left side and answer questions, the right side dynamically changes in response to your answers.

B. *Using WinDraft Data Entry: An Overview*

1. Load Word.
2. Load WinDraft--simultaneously press <Ctrl> <Alt> <W>.
3. Click on the Change Practice System icon and choose the system heading that contains the documents you want to generate.
4. Click on the Checklist button to bring up WinDraft Checklist.
5. Optionally, use File Open to load the data from an existing master or client data file to avoid re-answering questions that will be the same for this client.
6. Answer the onscreen questions and, if applicable, make the appropriate changes to the data to accord with the new client's information.
7. Print the Checklist with the answers you have filled in.
8. Under File, choose Save As and supply a new data file name.
9. Click the "Blast Off" button and select the documents you wish to create.
10. Watch WinDraft assemble and merge the information to create your new personalized document.

The above steps are also explained in a little more detail in the Quick Start Chapter.

C. *Terminology*

1. Practice System

The practice system contains any group of related legal documents, such as all those associated with estate planning (wills, trusts, *etc*.), combined with the checklist screens, Word Templates, and customized "blast off" macros that support them. Practice system set-ups can be totally customized to match the needs of the firm or department and can mirror the existing organizational structure.

2. Model Document

The model document is the master form (*i.e*., the form from the form book). This is the foundation on which your customized documents will be built. The model document contains the variables and the instructions for inserting legal clauses. See the Author's Guide for a description of how to create model documents, and see the samples provided for specific examples.

3. Data Entry Screen -- The Checklist

The data entry screen or the checklist is a combination of the visual outline, the client and legal questions. It is created using a special scripting language described in the Author's Guide. The figure on the previous page shows the data entry screen.

4. Data File

The entered individual client information that goes into a document (*i.e*., client name and address). The data file is the file that results when the checklist has been completed and saved.

5. Merged Document

The final result -- an integration of the data file with the model document. This file can be saved on disk, printed, or edited in exactly the same way as any other Word document.

III. Using WinDraft

The following are the step-by-step procedures for using WinDraft.

A. Load Microsoft Word

Load Microsoft Word in the usual manner.

B. Load WinDraft

Click on the LoadWinDraft button or press the **Control-Alt-W** keys simultaneously to load the WinDraft global add-ins. You will see the WinDraft toolbars appear. (If you don't see the Load WinDraft button, you can select it, under the heading "More Toolbars," by clicking the right mouse button while the cursor is on any toolbar.) There is no startup screen for WinDraft. It functions as an add-in to Word.

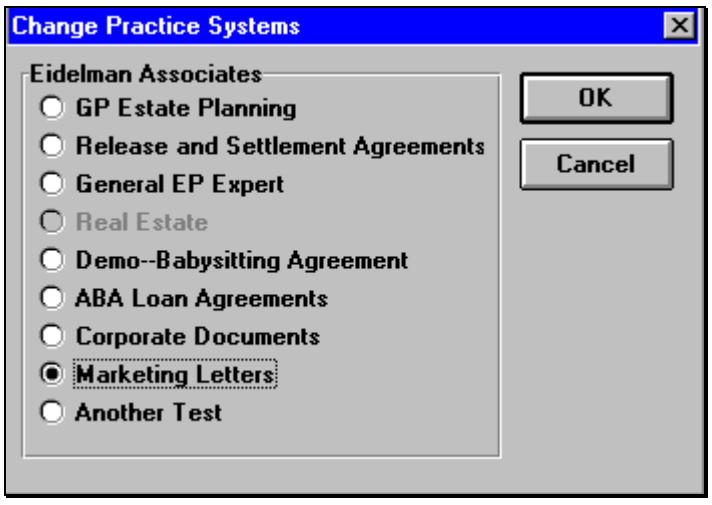
C. Select Practice Systems

1. Practice Systems Button

Click the Practice Systems button

This button appears in the WinDraft toolbar on the upper part of your screen and is used to switch between different Practice Systems.

If only one Practice System exists this button will not be used.

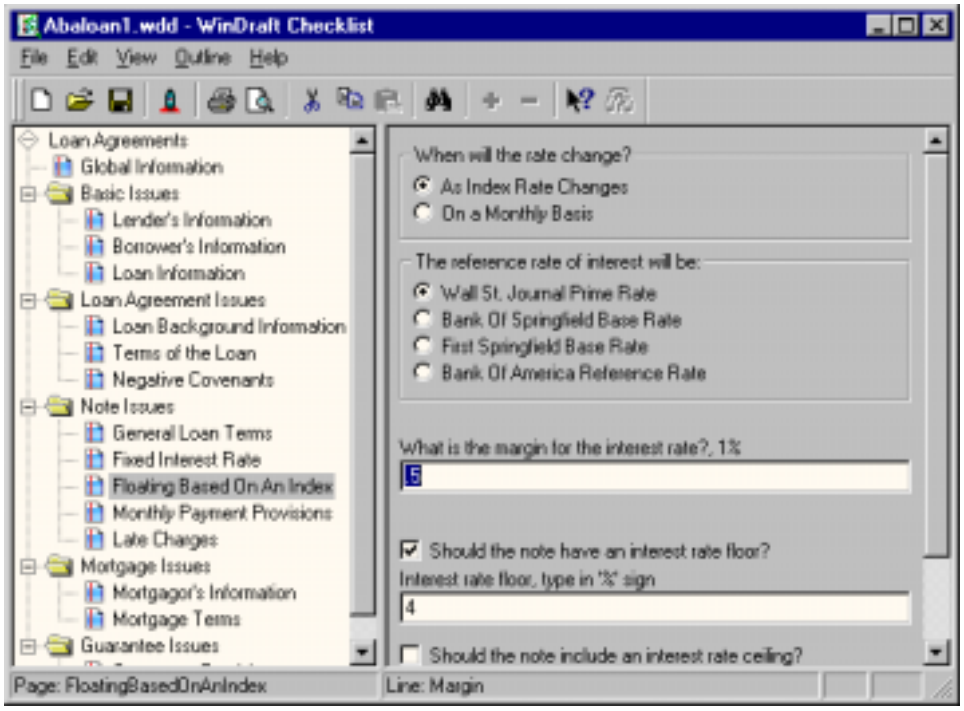


Your systems will vary, depending on what you or your staff have set up on your network or PC. Select the system you want to use. (If you only use one system, don't worry about it. The last system you used will remain as the default.)

2. "Checklist" Button

Checklist Button

Clicking on the toolbar's "Checklist" Button automatically loads WDData, the WinDraft data entry program. The data entry screen appears, and client information can be applied to legal questions.



3. Exploring the Outline

Complex practice systems, such as estate planning, can have checklists that exceed 20 pages. It is cumbersome to navigate through all of this information with paper checklists and other conventional approaches. It is easy to get lost.

To alleviate this problem, we organized the data entry screens in outlines. As shown, the outline appears on the left side of the screen, while the legal questions and client information appear on the right.

When the Data Entry checklist loads, the outline is in its expanded form, meaning that all folders and subfolders are shown. By clicking on the minus sign next to a folder, or by selecting Collapse from the Outline menu, you can hide all the subfolders of a folder. Conversely, by clicking on the plus sign or choosing Expand All Children from the Outline menu, WinDraft will open up all of the folders and subfolders within the currently selected folder. Thus, if this is clicked on when you are at the first folder, the entire checklist is displayed.

To move through the outline, either click on the folders and pages on the left side of the screen, or use F4 (Outline | Find Next Unanswered) to move to the next unanswered question. Ctrl+Tab will move the keyboard focus between the left and right side of the screen. Tab will move you down to the next question on the screen, while Shift+Tab will move you back, to the previous question. Ctrl+> will move you to the next page on the outline, while Ctrl+< will move you back to the previous page.

There are three symbols that are used to facilitate the viewing of the outline. They are also indented and connected by lines to show their relationships:

Closed Folder. There are sub-levels that are closed. You can open them by clicking the mouse on the folder, by pressing the "+" key on the numeric key pad, or by using the mouse to click on the "+" button on the tool bar.

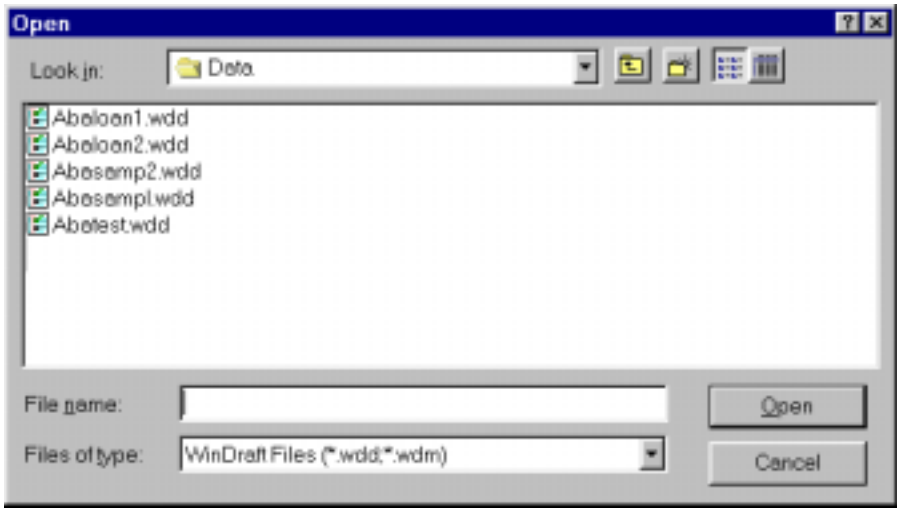
Opened Folder. This level of the outline is open, so that you can see the screens or sub-folders contained within this folder. You can close it, if you wish, by clicking on the folder, or by using the "-" key on the keypad or clicking the mouse on the "-" button on the tool bar.

Page of Paper. Data entry screens are locations into which the user inputs client information, in response to WinDraft's questions. A data entry screen will be displayed on the right side of the screen when this item is selected, and there are no subfolders or pages.

IV. Entering Client Information

Client information can be entered by two different methods: (1) from scratch, using "File/New," or (2) if the client's information will correspond closely to an existing client's data or a master, by loading that master client data file and making the appropriate changes. A veteran user will establish his or her favorite answer patterns to correspond to most of the factual patterns that are encountered. When a new matter arises, the appropriate master client answer file is pulled up and only the client-specific changes need be entered. This results in great efficiency -- one of WinDraft's key benefits. In normal operation, you would always start with a master data file or an existing client file and make appropriate changes.

To open a client's file, (1) use File Open or click on this toolbar icon to open a data file; then, (2) select the appropriate client's data file. If you are using DOCS Open, iManage or another document management system, the interface will be the usual DMS profiles.



A. File Names

WinDraft's data files can have any valid name DOS allows, but we strongly suggest you use names that end with WDD or WDM, according to the following convention:

*.WDM	Master Data Files
*.WDD	Client Data Files

Use WDM for "master" data files that contain standard answers that can be used to save time.

Use WDD for "client" files that relate to a specific client or test. (Sam Jones' data file might be named JonesSam.WDD)

Loading a master data file or a different client's file with similar facts let you avoid re-entering all of the data for each client. Each time you work with a new client, load a Master Data File or data file completed for a previous client and then change the items that are different. Then use File Save As to rename a data file as a client data file. (**DO NOT use File Save**, unless you want to overwrite the previous data file.)

For example, in the estate planning context, if you frequently use a particular set of options with a Generation Skipping Trust, you might name it GSTrust.WDM. Then, each time you draft a Generation Skipping Trust, you can Open GSTrust.WDM, change the data as needed for that particular client and Save As a new name, using the client's name and the .WDD extension. This saves time; there is no need to re-answer non-client specific questions.

B. Answering Questions

Edit the data to suit your needs, making sure to answer all the relevant questions. Follow the outline-processing interface.

To move from question to question on the data entry screen, use the tab keys, the keyboard or the mouse. By pressing Tab, you will proceed to the next selection. By simultaneously pressing Shift+Tab, you will return to the previous question. F4 or the selection "Find Next Unanswered," found under the Outline menu, can also be used. This function enables the user to jump directly to those questions that have not yet been answered and which need to be answered. This command is very helpful in making sure that all appropriate provisions get included in the final document.

If you have questions about what is meant by a question, or would like to review the law concerning a question, then press Ctrl+F1 or choose the menu item Help | More Information. This will bring up context sensitive help about the particular page and question you have selected. Not all practice systems support this: if the Help | More Information menu item is grayed out, then the author of your system has not yet written help screens.

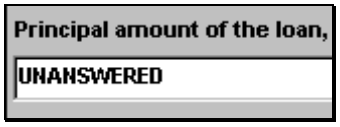
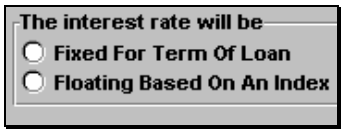
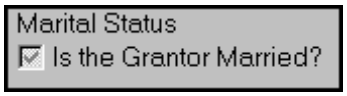
If you wish to return an answer to an UNANSWERED state, use the Unanswered option under the Edit heading (Ctrl+U).

You will note that irrelevant data questions are "grayed out." For instance, in the ABA Loan Agreement System, if you input that there is only one lender, the questions applying to multiple lenders and their information will be displayed in gray on the screen and cannot be selected. This prevents erroneous provisions from being included in the final document and avoids wasting the user's time in maneuvering past irrelevant questions.

If a question is grayed out but is answered incorrectly, you must change whatever answer is causing it to be grayed out in order to change it. Thus, for an Estate Planning document, if you start with a data file that says the client has children and the author wants to change it for a single person without children, you will

need to first change “subsidiary” or dependent questions to be answered, then change the answer stating the client has children to “No.”

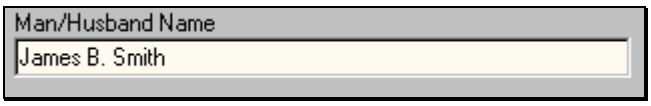
Unanswered questions in the checklist are indicated as follows:

Text boxes	“UNANSWERED”	
Radio Buttons	None selected	
Check boxes	Shaded Gray	

Be sure to answer all relevant questions. When you have answered all of the questions for that section, WinDraft automatically returns you to the first question. At this point, verify that the information you entered is correct or proceed to the next section of questions. This is accomplished by using the mouse to move to the left side of the screen and choosing the next icon in the outline or by hitting F4.

There are several kinds of controls used in the WinDraft Checklist to enter information. They work the same way as they do in other Windows programs.

1. Text Boxes – Normal, Long, and Masked

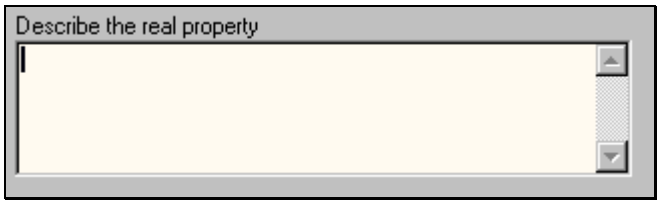


In response to a text box, simply type in your answer. These answers can contain up to 8000 characters. If you type beyond the right side of the text box, the data scrolls like a ticker tape. Even if it is not visible, the data is all there. The <Home> and <End> keys will move you to the start and end of the text, respectively.

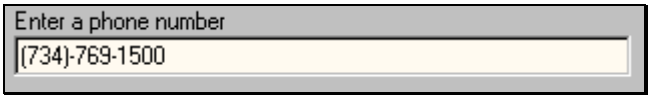
If the question is unanswered, it will say “UNANSWERED” in the checklist.

There are two variants of text boxes.

Long text boxes work in exactly the same way as normal text boxes, but let you see more of what you have typed. Long text boxes also accept returns (when you press the “Enter” or “Return” key inside of them).



Masked text boxes look very much like normal text boxes, and accept keyboard input in a similar way, but they only allow information that is formatted a certain way. Masked text boxes are often used to enter phone numbers, dates, currency amount, and so forth.



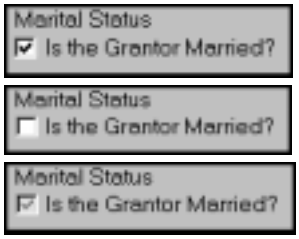
2. Check Boxes

Check boxes represent questions calling for a Yes/No response. They can have three states:

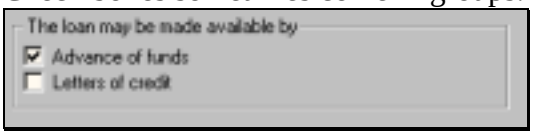
1. Checked, which means Yes

2. Unchecked, which means No

3. Hatched, meaning it's Unanswered

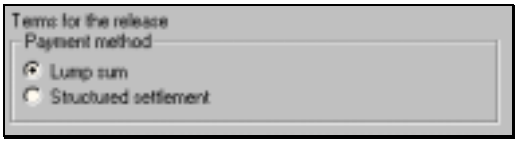


Check boxes sometimes come in groups.



3. Choices – Radio Buttons and Dropdowns

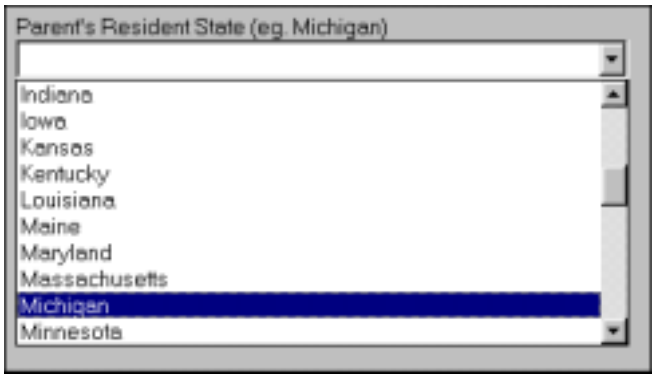
Radio buttons are used for multiple choice options. Only one option can be selected. If one choice is already selected, click on another to de-select the first choice.



If the question is unanswered, no button will be selected (they will all be white).

For questions which have more choices than are convenient to display in the radio button format, dropdown lists are used. A drop-down box is similar to radio buttons in that it offers a choice of one item from among many alternatives. It differs, however, by taking up much less "screen real estate" when not in use. To make the indicated selection, press the drop-

down button, use the scroll bar to bring your answer in view and click on the answer. If the question is unanswered, the drop down box is empty, like an unanswered long text box.

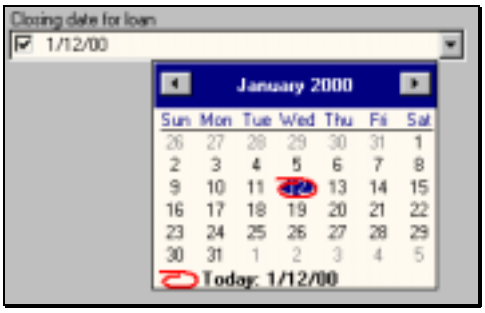


There are two types of drop-down boxes supported in WinDraft. One that limits your choice to the items in the drop-down list and one that allows you to either make a selection from the list or type in text of your own. The latter acts like a textbox when you tab to it, but gives an arrow at the right that you can click to fill in a predetermined choice.

Dropdown lists may be specified in advance by the author of your practice system, or may come from a database.

4. Dates

A calendar control may be used for entering dates. This looks similar to a dropdown box, but it has a box at its left which is checked when the control has been answered. Clicking the arrow at the right of the textbox drops down a calendar.



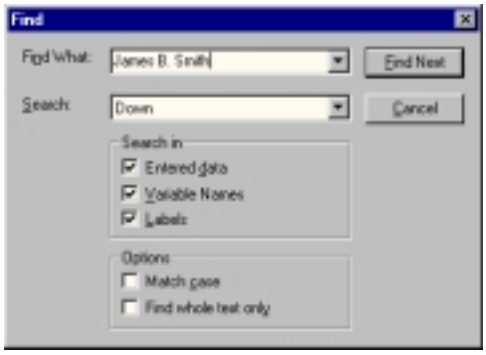
5. Buttons

Button controls, when pressed, activate programs written in VBScript or JavaScript. There will usually be labels above them indicating what they do.

C. Advanced Navigation

1. Searching for Questions

Often WinDraft is used to design very complex systems, and while the outline interface makes it easy to navigate, sometimes one forgets where a particular legal option is set. WinDraft Checklist includes a full featured Find feature for just these times.

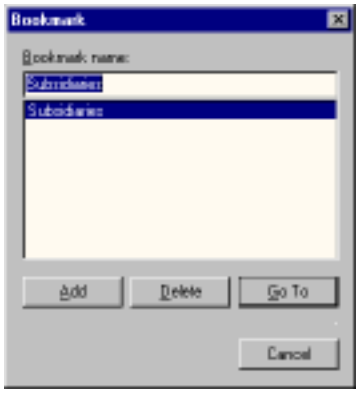


Press Ctrl+F or select Edit | Find from the menu. Enter the text you want to search for, and choose whether you want to look in the information you typed in, or in the labels designed by the system author. (If you are a system author, you might also want to search for a variable name).

WinDraft will remember your selection the next time you bring up the Find dialog. Also, to repeat your last search, press F3.

2. Bookmarks

If you would like to remember where a particular question is so that you can come back to it later, you can use WinDraft's bookmark feature. Simply choose Edit | Bookmark from the menu, type in a name for the bookmark, and click add. Later, when you want to return to the bookmark, choose Edit | Bookmark, select your bookmark by name, and click Go To.



Bookmarks are saved with your WinDraft Checklist document, and can be returned to in later sessions.

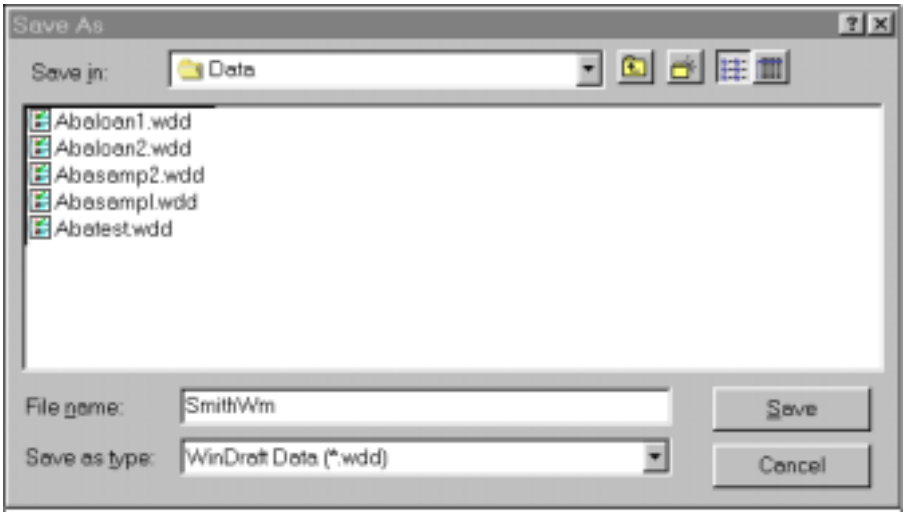
V. Saving Client Files

A. Saving WinDraft Files

WinDraft utilizes the same buttons and menu choices that are typical of all programs in Microsoft Windows. File New, File Open, File Save As, and File Save work just as they do in Microsoft Word or Microsoft Excel.

File Save As will save your modified data file, using any name you want of up to eight characters. The file is saved as ".WDD" for client data or ".WDM" for model data. If you are not using document management, we suggest using the client's last name as the file name so that the appropriate file can easily be recognized. If you want to save under the same name, use File Save, or click on the Save button.

When using the File Open and File Save As commands, you will get a dialog box that looks like this:



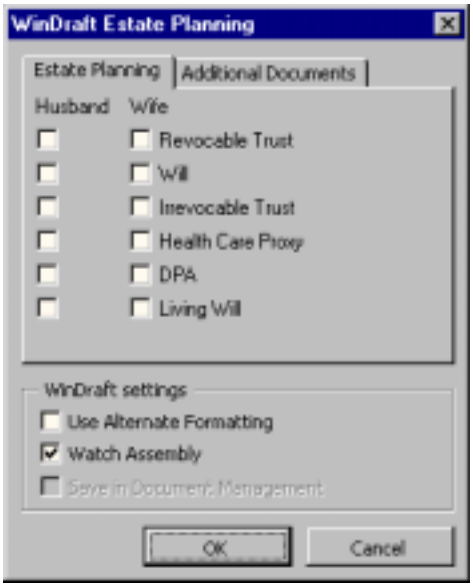
If you forget to save your file, WinDraft automatically asks you whether you want to save it when you choose the Blast Off! icon (discussed below) or when you exit the program. If you answer "yes," the above screen appears. This is equivalent to a File Save command. WinDraft overwrites the old file with the new data but it retains the same file name. To avoid overwriting the existing data file, choose "Cancel" then do a File Save As. This gives the file a new name and avoids overwriting the old file. The files should be stored in a directory labeled "data."

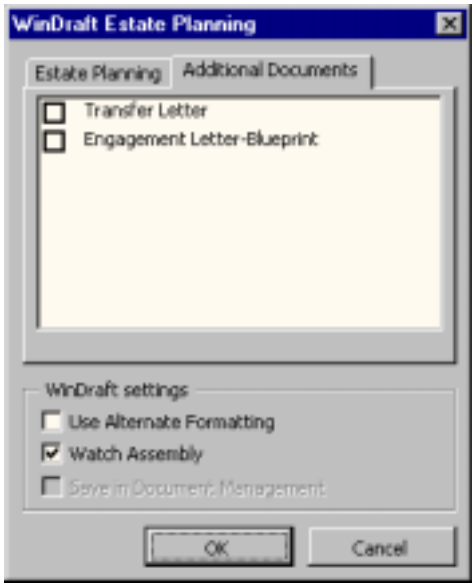
VI. Blast Off To Merge New Documents

A. Merge (Blast Off!)

Blast Off, which is represented by this rocket icon, is the function that takes the client information you entered in the data file and merges it with the model document to create a customized legal document.

By clicking on the rocket icon, the user accesses the "Blast Off" dialog box. For the estate planning system, the dialog box looks like this:





Select the documents you wish to create. This box can be customized by the firm and by the department to show all the related documents that may by merged. Customizing the Blast Off dialog box is described in an Addendum to the Author's Guide: PROGRAMMING CUSTOM BLAST-OFF DIALOGS. WinDraft can produce as many documents as you wish to program. Documents that do not apply may be grayed out if the system author so chooses.

1. Select Document Format

There are several options you may select at "blast-off" time:

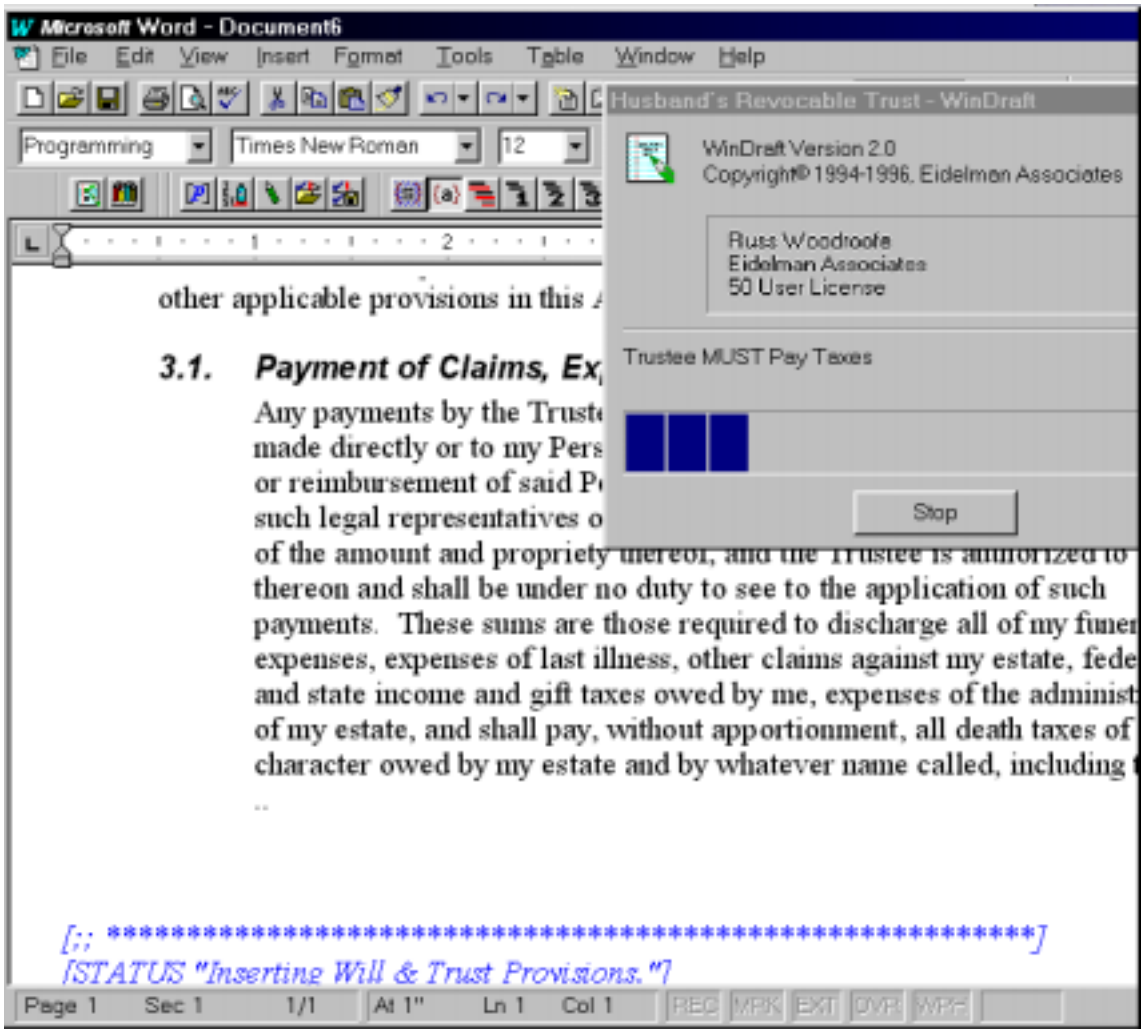
Format Settings – (Alternate Formatting) WinDraft can change the formatting of output documents by clicking this button, in a practice system designed for such use. This allows your law firm to meet the needs and preferences of different members.

"Watch Assembly" -- Although it will be slightly slower, we recommend that you click on "Watch Assembly." Enabling this feature allows the user to watch the document being assembled on screen.

"Save in document management" – If you are using a supported document management system (currently DOCS OPEN or LawOffice), you will be given the option to have the documents automatically saved in the document management system with profiles automatically filled out. Answer "Yes" if this is a "final run," and "No" if you are just testing. If you are not using a supported document management system, this option will be grayed out.

2. Assembly

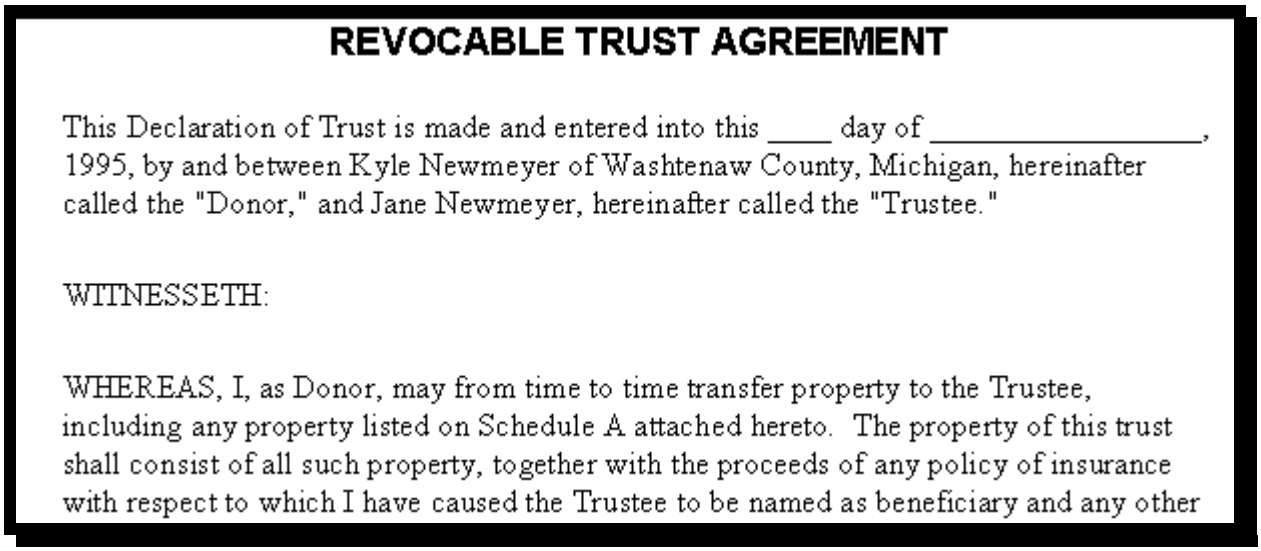
When "OK" is pushed, WinDraft will merge the model documents and the client data you entered to create, on the screen, the selected documents. During the merging process it displays a progress dialog box that shows the percent (%) completed and status information. The status information tells you what WinDraft is doing with the document. A sample screen looks like this:


Microsoft Word - Document6
File Edit View Insert Format Tools Table Window Help
Husband's Revocable Trust - WinDraft
Programming
Times New Roman
12
WinDraft Version 2.0
Copyright© 1994-1996, Eidelman Associates
Russ Woodroofe
Eidelman Associates
50 User License
Trustee MUST Pay Taxes
Stop
other applicable provisions in this
3.1. Payment of Claims, Ex
Any payments by the Truste
made directly or to my Pers
or reimbursement of said P
such legal representatives o
of the amount and propriety thereof, and the Trustee is authorized to
thereon and shall be under no duty to see to the application of such
payments. These sums are those required to discharge all of my funer
expenses, expenses of last illness, other claims against my estate, fede
and state income and gift taxes owed by me, expenses of the administ
of my estate, and shall pay, without apportionment, all death taxes of
character owed by my estate and by whatever name called, including
[;; **********]
[STATUS "Inserting Will & Trust Provisions."]
Page 1 Sec 1 1/1 At 1" Ln 1 Col 1 REC MPK EXT OVR WPH

When WinDraft is finished, you will see a "Document Assembly Completed!" message and you can save and edit the documents like all other Word documents. (If you clicked the "Save in document management" checkbox", the system will create a profile for each document and display a "hit list" of the documents WinDraft just created.) WinDraft automatically fills out the Word profile with

information present in the document and data entry screens, or, if you are using DOCS OPEN, automatically saves it to document management. WinDraft can also automatically output tasks, journal entries, and other items to Outlook, or do other interfacing with case management and other systems if your practice system is configured for it.

You have now created your customized document in just a fraction of the time it used to take, and with fewer errors.



REVOCABLE TRUST AGREEMENT

This Declaration of Trust is made and entered into this ____ day of _____, 1995, by and between Kyle Newmeyer of Washtenaw County, Michigan, hereinafter called the "Donor," and Jane Newmeyer, hereinafter called the "Trustee."

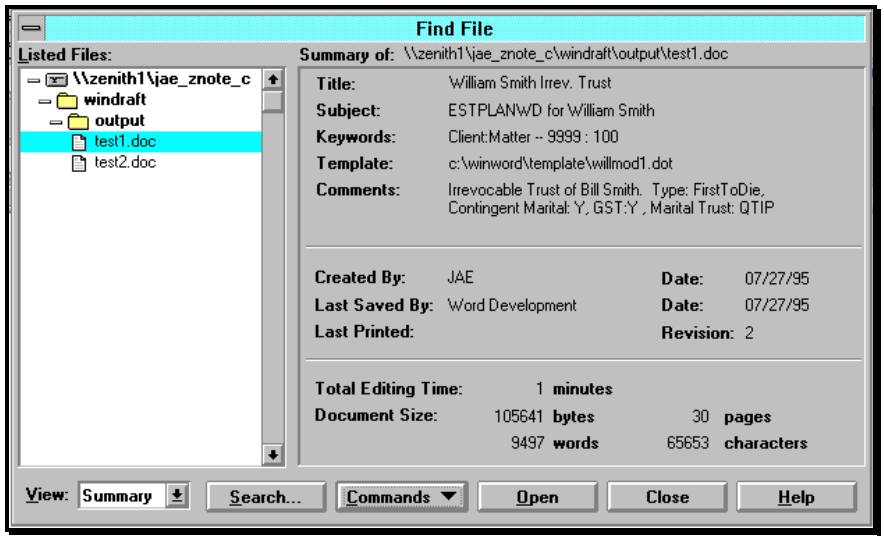
WITNESSETH:

WHEREAS, I, as Donor, may from time to time transfer property to the Trustee, including any property listed on Schedule A attached hereto. The property of this trust shall consist of all such property, together with the proceeds of any policy of insurance with respect to which I have caused the Trustee to be named as beneficiary and any other

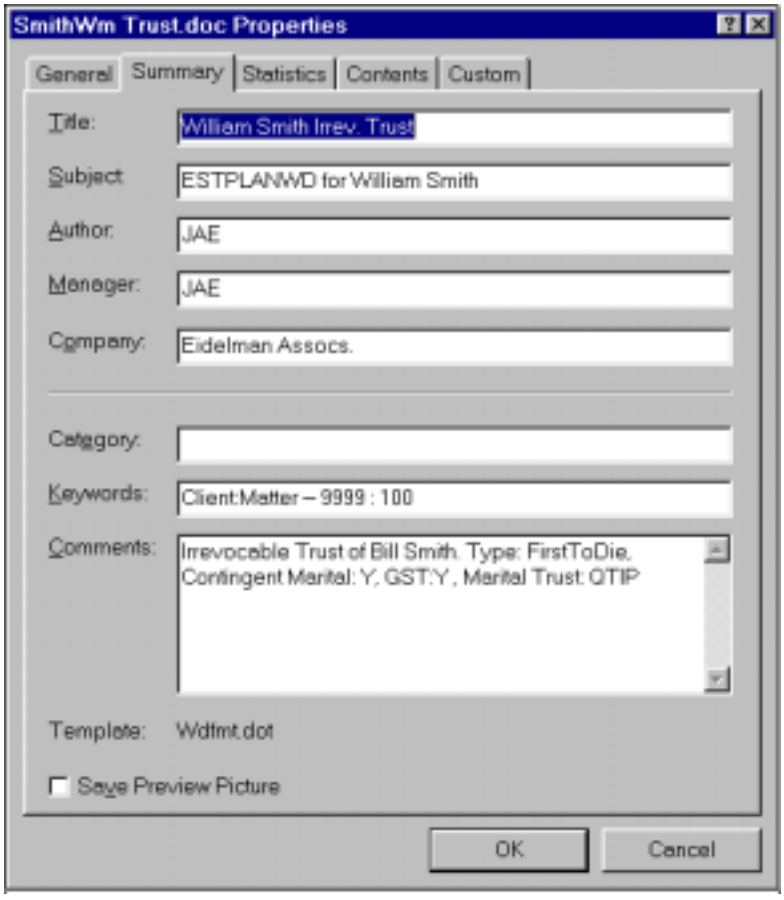
3. Working with Automatic Document Summaries

When Word saves a document, a few items of key information are saved along with it. In WinDraft this summary information may be customized and rendered more useful by including a PROFILE command in the model document (see Author's Guide, Authoring Model Documents) . If you use Word to manage your documents, the PROFILE command will output information about the author, subject, title, and description to the Word Document Summary. If you use DOCS OPEN, the profile information will automatically fill out the DOCS profile. These automatic document summaries will make your documents easier to retrieve and manage.

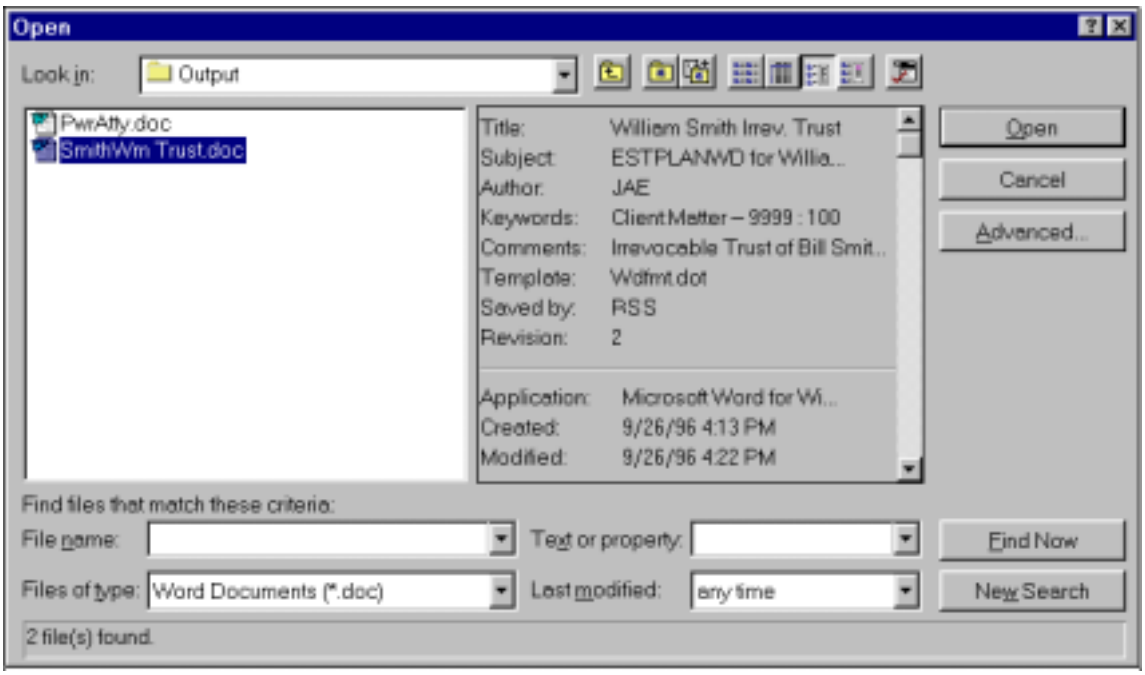
You can access this summary information from Word in different ways, depending on which version you are using. In Word 6.0, you may select File|Summary Info to obtain information on the current file, or you may choose File|Find File. Using File|Find File, you can search for documents by author, client/matter number, document type, name and description/comments. Here is what you will see if you use File|Find File in Word 6.0 to search with the "Summary" view turned on:



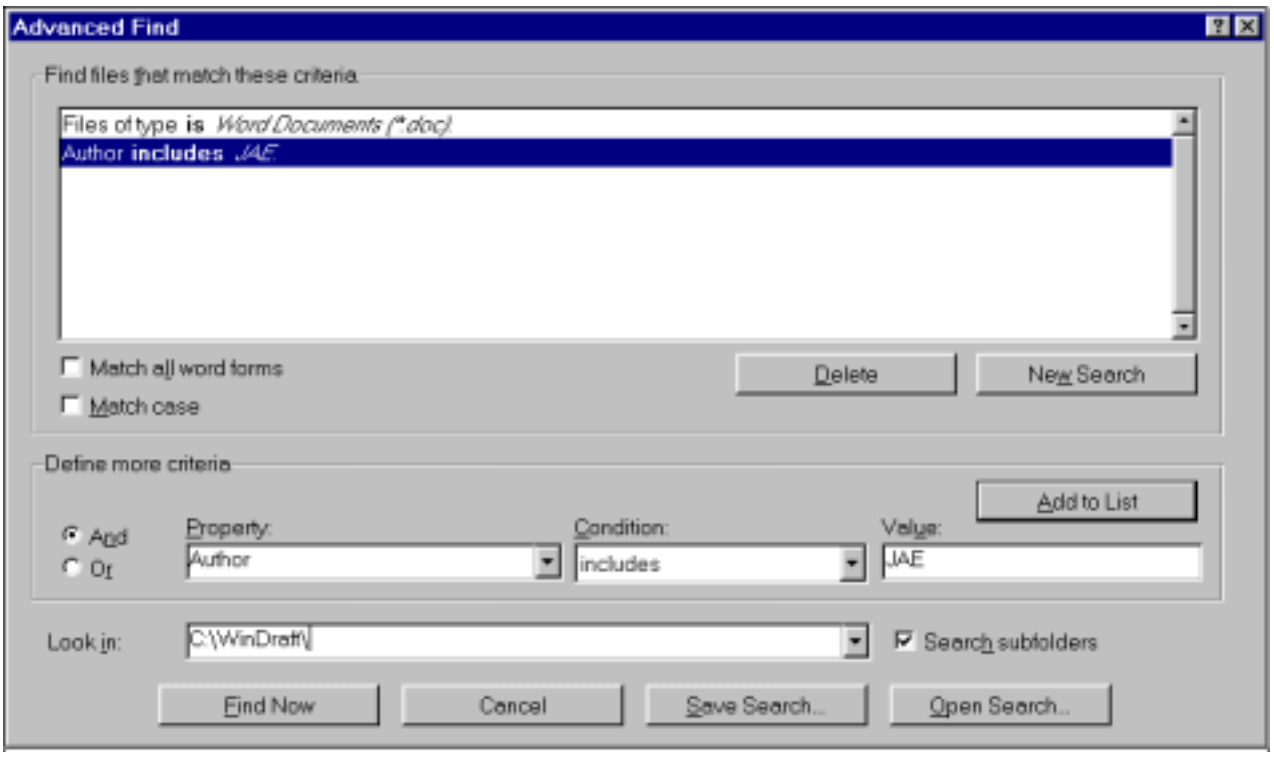
In Word 7.0 and later, if you want to see the summary information for the file currently open, choose File|Properties and then click on the Summary tab to get the following view.



On the other hand, in Word 7.0 and later, you may also select File|Open. Highlight the file of interest and, if you click on the Properties button in the upper right, you will see the summary properties displayed in the right-hand panel, as shown in the following figure.



You may search for a file by any of the fields (Title, Subject, Author, Keywords, Comments, etc.) listed under properties. To do so select Advanced and, in the Advance Find window (see the figure below), select

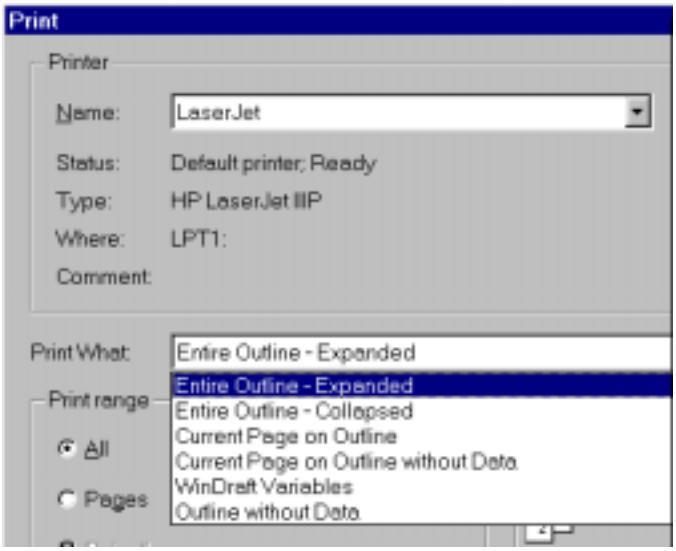


the Property you wish to search by and the Value you are seeking and press Add to List to add that criterion to your search. Then choose the scope of the search in Look In at the bottom and press Find Now to execute the search.

VII. Printing

A. Printing WinDraft Data

To print your data, click on the Print icon or select File | Print from the WinDraft Checklist menu. If you want to review the print output on the screen, you can select File Print Preview. Note that if you print from the data entry screen, there are six ways to print your data:



1. Outline -- Expanded

Normally, print the expanded outline to see your data presented in a "lawyer-friendly" way. **We recommend putting a copy of the data outline printout in the file** so that, as the documents are proofread, a lawyer can verify the selected options. This saves a tremendous amount of time. Instead of the attorney proofing an entire document he or she can now just review the checklist forms to make sure that the appropriate boxes were checked. If they were, then the provisions should be in the document

This option prints all of the information on all of the data entry screens, both the prompts and the created data. Standard prompts and data will appear in black, indented to the right outline level. Formatting of the font indicates special information as follows:

Font	Type of Information
Bold and Underlined	Headings
Light gray	Inapplicable/Irrelevant
Italic	Unanswered

2. Outline -- Collapsed

This option prints just the outline from the left side of the data entry screens.

3. Current Page

Expanded -- This option prints just the prompts and filled-in data from the current data entry screen displayed on the right side of the screen.

Without Data -- Same, but with blanks for the data.

4. Variables

This option prints all the variable names and their values (the answers). This is useful for debugging a new practice system.

5. Outline Without Data

This option prints a blank outline, for use by lawyers who are not on the system. Computer-phobic lawyers, also, can use WinDraft on paper while their secretaries input the data on screen.

B. Printing the Final Document

After it is merged, the customized document can be printed by selecting Print and carrying out the same steps you follow to print any Word document.

VIII. Sharing Information With Other Users

A. Emailing WinDraft Data

To email the document you have open to another user, go to the File | Send menu item. A new message will come up in your email program, with the current data file attached. Fill out the To: and Cc: fields, edit the message to explain why you are sending the document, and send it normally.

B. Emailing the Final Document

After it is merged, the customized document can be emailed to another user by going to File | Send To and carrying out the same steps as used to email any other Word document.

IX. Formatting in Word

A. Styles and Templates -- The Concept

When WinDraft creates a new document, it attaches a .DOT file and will use formatting based on styles in that .DOT file. See your Microsoft Word documentation for more information on styles and how to use them.

There is an option, Reset To Styles, in the WinDraft.INI file that tells WinDraft whether to respect formatting in newly merged documents that is based on the model document's formatting, or to reformat based on the styles in the .DOT file. See Appendix 3, WinDraft.INI File Settings, for further information.

The functions described in this section are simply optional ways we use Word formatting, rather than proper parts of WinDraft, except for the additional toolbars and styles we have created. You can use any of Word's versatile formatting functions to format WinDraft documents.

B. The Edit Tools Toolbar

WinDraft is extensively integrated with Word and supports the use of MS Word bookmarks and fields, automatic paragraph numbering, table of contents, cross referencing, and outline headings. In fact, many of our model documents extensively use these features.

The Edit Tools tool bar contains buttons which apply styles to text, prepare tables of contents and permit the viewing of field variables.

If the special Edit Tools toolbar is not visible, select the View | Options menu item from WinDraft Checklist. This will bring up a dialog, which includes a WinDraft Editing Toolbar setting. Make sure that the latter is checked, and then press the OK button; the Editing Toolbar should now be displayed. This selection will "stick" each time you change it and enter or exit WinDraft.

1. "Field" Variables

Microsoft Word "Field Variables" are used to dynamically change gender-related terms (his/her, husband/wife), set subject-verb agreement, apply appropriate document-type terms (trustee/personal representative), *etc*. You can use two special format display buttons to view these fields in different ways.

a) View Field Codes

☐ The “View Field Codes” button will switch fields back and forth (toggle) between a display that shows the calculated value of the field, such as “husband,” and a display of the Word field codes themselves, such as {REF HusbandWife}. This feature greatly increases the readability of the model documents for the reviewing lawyer. With this option, the text will read easily and will be easy to rewrite and modify.

b) Toggle Field Shading

☐ The "Toggle Shading" button will turn on and off the display of shading on the screen that highlights fields and makes them easy to pick out from ordinary text. This can be used whether or not fields or values of fields are displayed.

To further explain, the document can be viewed three different ways during editing, depending on the needs of the author/reader:

1. to my wife to be hers ... (as in a client document)
2. to my wife to be hers .. (as in a client document, but shaded)
3. to my **{**ref HusbandWife**}**, to be **{**ref SpouseHisHers**}** absolutely

By turning the “View Field Codes” “Off,” as in Options 1 and 2, the author of the model document can clearly read his or her provisions, see how they are used and avoid distraction.

(1) “Recalculating” the Document

WinDraft will automatically “recalculate” all of the fields, so generally the user will not need to be concerned about this.

(Background for debugging and manual operations: In order for the changes in fields to take place, the portion of the document containing those fields must be “recalculated.” This is done by selecting the text to be calculated and pressing [F9]. To select the whole document, put cursor in left margin and press Control-Mouse Click, press <Ctrl-A> (for “all”), or choose Edit|Select All.

2. Styles for Formatting And Outline Headings

Text within the document is formatted with “styles.” These styles control the appearance, but not the content, of the text. If you want to change the font, font size, or formatting of the text, it is done through selecting styles.

Within Word outlines, there are predefined styles for nine levels of outline heading: *Heading 1*, *Heading 2*, *etc*. These styles control the structure in outline view and the font, indenting, *etc*. in normal view. They also have

additional attributes in normal view to control paragraph numbering, table of contents, cross reference, *etc*.

Note that Heading styles are defined automatically by Microsoft Word as paragraph styles. This means that they will apply to any text from the beginning to the end of the paragraph. A paragraph ending is marked with a paragraph" mark, which, when you select Tools - Options - View - Show Non-Printing Characters -- Paragraph Marks, looks like this icon: ¶

A heading, therefore, *must* be on its own line! If you must have text that *looks* like a heading and *acts* like a heading, but has a normal paragraph in the same paragraph, there are workarounds involving more effort, but a *real* heading must be on a line by itself. This is a limitation of Microsoft Word.

3. Formatting Body Text

The first button automatically applies the appropriate *body text* styles for the entire document. Body text styles have been defined for four levels. For forms that are already formatted, you will usually only need to apply text formatting to the new text you type.

The Outline tool bar also contains four menu items for specific body text styles under the Body Styles toolbar item. By selecting these (the icons are shown above) the paragraph will be indented to the appropriate level. If you are at Outline Level One, or wish to indent the text one level, use the button labeled "**1,**" at Outline Level Two, use the "**2**" button, and so on. Each button progressively increases the paragraph's indentation. Type the text first, then place the cursor anywhere within the paragraph (or highlight multiple paragraphs), then click the appropriate button. This function ensures that the same font will appear throughout the document.

These buttons will work properly with styles defined in the template WDfmt.dot. WinDraft applies this template automatically to all merged documents. Other Word documents may not define the same body text styles, and the buttons may not work correctly or (if that style is not defined at all) may simply beep and do nothing.

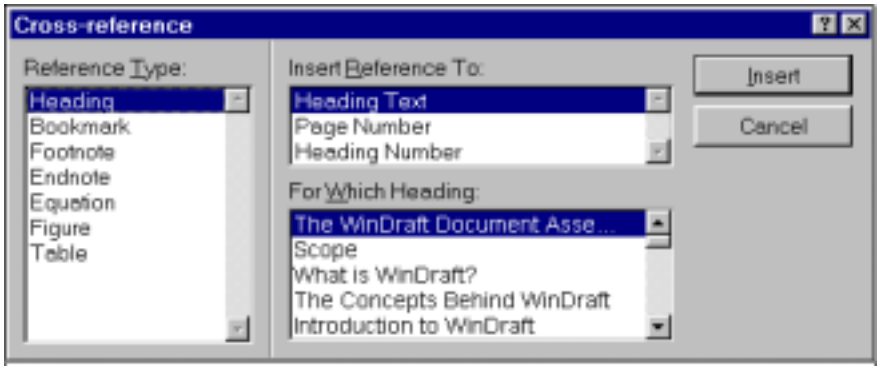
(a) Automatic Paragraph Numbering

We are using Word's automatic paragraph numbering. The best way to change the automatic numbering style is through the menu sequence Format|Style|Modify|Format|Numbering.

For a given Document, you can also use Format|Heading Numbering.

(b) Cross-References

Cross-references are added in the form of Word fields. To insert one, use Insert cross-reference and select Heading Type and Heading Number on the dialog box that looks like this:



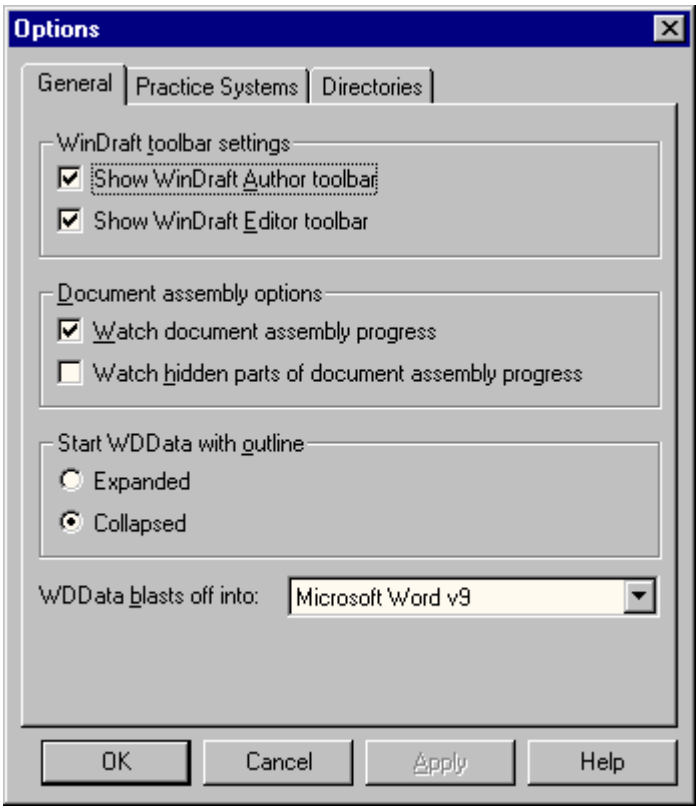
When you look at the cross-references with the "Show Fields" button pressed, you will see something like ***{ref REF9944558833}***.

4. Table of Contents

☐ A table of contents can be generated for your documents merely by pressing the “TOC” button. The table of contents, which will be inserted at the end, is automatically customized to fit your newly-created document.

X. WinDraft Options

WinDraft has a number of user adjustable settings. If you would like to adjust some of these, choose the View|Options menu item from WinDraft Checklist. This will present a dialog box which looks like this:



These checkboxes control WinDraft as follows:

A. WinDraft Toolbar Settings

1. WinDraft Author toolbar

This setting decides whether the WinDraft Author Toolbar should be displayed or not. Most users should leave this checkbox unchecked, unless they are editing model documents. See the Author's Guide for more information.

2. WinDraft Editing toolbar

This setting determines whether the WinDraft Editing Toolbar should be displayed or not. Most users should check this if they plan on editing the formatting of their output documents.

B. WinDraft Default Settings

1. Watch document assembly

This item determines whether the Watch Document Assembly checkbox in the blast off dialog boxes should be checked by default.

2. Watch hidden parts of document assembly

If this option is checked, then DISPLAY OFF commands are ignored during document assembly. Note that this has no effect if Watch Assembly is turned off.

XI. How to Read Model Documents

WinDraft model documents are Word 6.0 documents with two types of "programming." These use the built-in functionality of Microsoft Word, and the WinDraft add-in to insert text and variable information.

With some practice, anyone with the requisite substantive knowledge for that area can create their own documents by using WinDraft's concise vocabulary of programming commands (there are less than 20 commonly used commands) . This section is intended to teach only the basic concepts about WinDraft model documents. For more detailed information, and to create your own model documents, see the Author's Guide.

A. WinDraft Syntax and Concepts

There are two primary functions that WinDraft performs:

1. Inserting or "Merging" a Variable

To insert a variable in the text, simply insert the variable in the text of the model document, with braces around it, like this:

> This contract is entered into between [Party1Name] and [Party2Name].

2. Conditional Text or "Rules"

"Rules," which govern whether optional or conditional text is inserted into the customized document, are represented in the text by "IF" statements. As with all programming phrases, they are embedded in square braces, like this:

> *[IF PrincipalDistribution = "Ascertainable Standard"]*
>
> The Trustee may, from time to time, pay to or apply for the benefit of my wife such amounts of principal as the Trustee deems necessary or advisable for her health, education, support and maintenance.
>
> [*ENDIF]*

Alternatives may be accomplished by using the "ELSE" command, like this:

[IF GrantorState = "Minnesota"]
Minnesota Statutes Section 525.528 shall not be applicable to the allocation of assets to the marital share.
[ELSE]
Generalized text for other states goes here.
[ENDIF]

To make the programming of rules easier to read, we have put them in italics, although this is not required, and you may use whatever formatting you choose including putting them in color. Most of the programming statements that stand on lines by themselves are formatted with the "Programming" style and can be looked at in any font you want with the change of a few changes to the style sheet. The Programming style is removed from the output document, to ensure that no accidental programming fragment find its way into merged documents.

3. Document Structure

a) Subroutines

The WinDraft command [INCLUDE "filename"] works to "nest" one model document within another at runtime. This makes it possible to break down a practice system into logical modules and subroutines which makes the system easy to modify, update, and organize. A model document author may wish to use this option to hide complex programming in another file or to store common clauses or provisions in their own program so that they can be easily called into action from new or existing model documents.

4. Estate Planning Documents

Some legal documents are comprised of a group of several smaller documents. On WinDraft, such a group is called a "Practice System." One example is the Estate Planning System. The modular structure of the estate planning documents follows the following general "nesting" to include lower level documents within higher level documents.

RT_HUSB -- Revocable Trust of Husband -- Sets document type and gender terms)
- TRUST (Contains beginning and end of trust documents)
  - COMMON (Clauses that are common to long Wills and Trusts)
  - COMPOW (Clauses common to POW and long Wills and Trusts)
    - QDOT, *etc*. (Clauses that are called by several places within the document)

WILLHUSB -- Will of Husband
- WILL (Contains beginning and end of will documents.)
  - COMMON (Clauses that are common to long Wills and Trusts)
  - COMPOW (Clauses common to POW and long Will and Trusts)

QDOT, *etc*. (Clauses that are "called" from several places within the document)

Special Subroutines that do complex programming are also INCLUDED, such as those that execute rules, formatting, *etc*. These are incorporated at various points.

B. WinDraft Model Document Command Summary

Command	What it Does	Example
[*variable*] or [expression]	Fills in the blank	I, [*GrantorName*], hereby ...

IF ... ENDIF
IF ... ELSE ... ENDIF
IF
ELSEIF
ELSE
ENDIF
	Rules for logic	[IF *FractionOfResidue* = "Yes"]
		Fraction of residue formula goes here.
		[ELSEIF *PecuniaryFormula* = "Yes"]
		Pecuniary Formula goes here.
		[ELSE]
		Something else goes here
		[ENDIF]

FOR … INCLUDE	Do something repeatedly	[FOR *I* = 1 TO *NumChildren* INCLUDE "ChildInfo.doc"] [FOR *I* = 1 TO *NumChildren* DO SET *ChildrenNames* = *ChildrenNames* + *Child(I).Name* + ", "]
INCLUDE	Incorporates another document. (equivalent to calling a "subroutine" in programming)	[INCLUDE "TRPOWERS.DOC"]
RULES	Same as INCLUDE, but for ASCII file containing only commands	[RULES "TAXRULES.RUL"]
GETDATA	Tells WinDraft to retrieve the information you entered at the data entry screen	[GETDATA]
SET	Set value of a variable	[SET *GrantorHeShe* = "she"]
ASK	Ask the user a question during document assembly	[ASK *CoverLet* "Include a cover letter?"]
FIELD	Inserts a Word field using the given text	[FIELD "DATE * MERGEFORMAT"]
;	Programming Comment (computer ignores what follows)	[;; JAE programming note: this is a note...]
LOG	Outputs an entry to a user-defined billing or other log file	[LOG "Timeslip.log", "JAE,12/2/95,JonesCorp,0029,2.5,Drafted asset purchase agreement"]
SYSLOG	Outputs an entry to the WinDraft system log file	[SYSLOG "Asset purch agmt drafted by " +Author]
STATUS	Displays message on status bar	[STATUS "merging " + *GrantorName* + " Trust"]
DISPLAY	Turns display of document merge on or off if Watch Assembly is turned on	[DISPLAY OFF]
PROGRESS	Sets the progress meter to the indicated percentage as the document is being merged	[PROGRESS "50"]
DOCTITLE	Displays document title in Windows title bar during merge.	[DOCTITLE "Loan Agreement"]
PROFILE	Outputs a profile string for Word or DOCS Open document summary properties	[PROFILE DocTitle = "Bill"]
GETDB	Reads information from a database	GETDB *Author*, DSN "ClientMatter", FROM "tblEmployee", GET "LastName, FirstName", WHERE "Email='" + *AuthorLogin* + "'"
HALT	Immediately stops document merge--used for debugging	[HALT]

Function	What it does	Example
SPELLOUT()	Spells out a number in text using Word functions, e.g., to convert 35 to "thirty five"	[SPELLOUT(*MinimumAgeForDist*)]
UC() or LC()	Upper case: converts "Andy Reynolds" to "ANDY REYNOLDS"; Lower case: converts it to "andy reynolds"	[UC(*GrantorName*)] [LC(*GrantorName*)]
IC()	Initial case: changes "some phrase" to "Some phrase".	[IC(*GrantorName*)]
PC()	Proper Case: changes "personal representative" to "Personal Representative"	[PC(*GrantorName*)]
REF()	References Word table, field or bookmark	[REF("TrustWill")]
MERGEFIELD()	Gets a Word Mail Merge Field	[SET *LastName* = MERGEFIELD("LastName")]
ADD, SUBTRACT	Add, subtract, multiply or divide 2 numbers (stored as string).	ADD("2", "2")
MULTIPLY, DIVIDE		
INC, DEC	Increment or decrement one number (stored as a string).	INC(*Count*)

LEFT, RIGHT	Take the first *n* characters on the left or right of a string.	LEFT(FirstName, 1)

Operator	What it does	Example
NOT	Works with IF to test if something is NOT true	[IF NOT *GrantorMarried* = "Y"]
AND	Compound logic	[IF *GrantorMarried* = "Y" AND *Children* = "Y"]
OR	Compound logic	[IF *GrantorMarried* = "Y" OR *Children* = "Y"]
=, <, >, C	Comparison operators (equal, less than, greater than, contains)	[IF *MaritalTrustCode* C "QTIP"]

See the Author's Guide for a full listing of commands, functions and operators and more complete explanations.

WDUREF5.DOC 2/18/97

WinDraft™

Expert Document Drafting Software
For Microsoft Word

Author's Guide

Eidelman Associates

Microsoft Certified Solution Provider

317 South Division
Suite 187
Ann Arbor, Michigan 48104

734-769-1500 (Voice)
734-769-1501 (Fax)
73067.2131@compuserve.com
support@lawtech.com
www.lawtech.com

WinDraft program, macros, field programming and Documentation Copyright © 1994-2000 Eidelman Associates, all rights reserved.

No part of this publication may be stored in retrieval systems, transmitted or reproduced in any way, including but not limited to photocopy, photograph, magnetic or other record, without prior agreement and written permission of Eidelman Associates.

WinDraft and EP Expert are trademarks of Eidelman Associates. All other brand and product names are trademarks or registered trademarks of their respective companies.

Printed 4/5/01

WinDraft™

Author's Guide

Organization and Master Table of Contents

The Author's Guide is actually an introduction and four documents on different, but related, subjects, on consecutively numbered pages as follows:

	Page
Introduction -- Simplicity is the Essence	1
Model Document Author's Guide	2
Checklist Author's Guide	50
Overview of Automating Your Forms	80
Programming Custom Blast-off Dialogs (Optional -- for more ambitious programmers)	87

WinDraft™

Author's Guide

Table of Contents

***I. AUTHOR'S GUIDE* INTRODUCTION -- SIMPLICITY IS THE ESSENCE 1**

II. MODEL DOCUMENT AUTHORS: WHO SHOULD USE THIS MANUAL 2

III. HOW TO USE THIS MANUAL 3

IV. CONCEPTS AND TERMINOLOGY 4

V. BASIC COMMAND STRUCTURE 5

VI. WORKING WITH VARIABLES 6

A. What Is A Variable? 6

VII. CREATING NEW SYSTEMS 8

A. To Create a New System: Using the Create New System Wizard 8

VIII. CREATING A MODEL DOCUMENT 9

A. Variables 9

B. SET 10

C. ASK 11

IX. BEYOND MERGING VARIABLES: LEARNING TO PROGRAM DOCUMENTS 13

A. [;; Comments -- The more the Better] 13

B. Basic Programming Concepts 13

C. IF Statements 15

D. INCLUDE 18

E. RULES 21

X. OTHER COMMANDS 22

A. GETDATA 22

B. DOCTITLE 22

C. PROGRESS 23

D. STATUS 23

E. SYSLOG 24

F. LOG 24

G. PROFILE 24

H. DISPLAY ON/OFF 24

I. FIELD 25

J. HALT 25

K. FOR 25

J. GETDB 26

XI. FUNCTIONS 27

A. UC, LC, IC and PC to Capitalize Letters 27

B. LEFT and RIGHT to break up strings 28

C. REF to Read a Word BookmarkField 28

D. MERGEFIELD to Read a Word Mail Merge Field 28

E. SPELLOUT to Spell Numbers 29

F. Math functions 29

XII. OPERATORS 29

A. NOT 30

B. AND 30

C. OR **30**

D. = and other Comparison Operators **31**

E. + to Join Text Strings Together **31**

F. Including Quotation Marks in String Constants **32**

XIII. WINDRAFT AUTHOR'S TOOLBAR 32

A. The Open Model Documents Button **32**

B. The Programming Style Buttons **33**

C. The Blast-Off Test Button **33**

D. Buttons to Work with Checklist Templates for WDData **34**

E. The Assemble Document Now on Screen Button **34**

XIV. AUTHORING ASSISTANT MACROS IN WORD 34

A. Insert Variable **34**

B. Insert Command **36**

XVI. UNDERSTANDING FORMATTING IN MS WORD AND WINDRAFT 37

A. How MS Word Defines Paragraphs **37**

B. MS Word Paragraph Attributes and Hard Returns **37**
1. Deleting a Single Hard Return 37
2. Deleting a Block that Includes a Hard Return 38

C. How WinDraft Handles Paragraph Attributes When Merging **38**
1. Returns After Closing Brackets May Migrate 38
2. Returns Within Command Brackets Are Lost Forever 39

D. How WinDraft Handles Character Attributes When Merging **39**

XVIII. TIPS FOR BETTER MODEL DOCUMENTS 40

A. How WinDraft Handles Whitespace **40**

B. Proper Placement of Brackets **41**

C. Combo Codes **42**
1. In General 42
2. Using Combo Codes in a Document 42

D. Math **43**

XIX. TIPS FOR BETTER CHECKLISTS 43

A. Handling Combo Codes in the Client Checklist. 43

XX. TROUBLESHOOTING MODEL DOCUMENTS AND CHECKLIST TEMPLATES 44

A. Template Changes Are Not Appearing in the Checklist 44

B. Runtime errors while merging 45
1. Spacing 45
2. Undefined Variable Referenced. 45
3. No End of Statement Bracket 45
4. Unmatched IFs and ENDIFs 46
5. Ending Quote Expected in Checklist Template 46

C. Unanswered 46

D. Formatting Problems 47
1. Extra Paragraph Returns 47
2. Indents and Other Paragraph Formatting Problems 47
3. "Body Text" Ends Up in a Heading 48

I. CHECKLIST AUTHOR'S GUIDE: WHO SHOULD USE THIS MANUAL 50

II. PURPOSE OF CHECKLIST GUIDE 50

III. CONCEPTS AND TERMINOLOGY 52

A. The Visual Outline 53

B. Authoring Checklists: command summary & examples 54

C. WinDraft Data Entry Screen 57

D. Saving Your Checklist Templates 57

D. Saving Your Checklist Templates 58

IV. BASIC CHECKLIST TEMPLATE COMMAND LANGUAGE 58

A. Asking Questions with WinDraft 58

B. Commands that Ask Questions and Get Data 59
1. TEXT 59
2. YESNO 63
3. CHOICE 63
4. CODE 65
5. BUTTONs 66

C. Commands to Format or Explain **66**
1. LABEL 66
2. TITLE 67
3. ; Comments 67
4. SPACE 67
5. LINE 67

D. LOGIC COMMANDS **68**
1. IF/ENDIF 68
2. SET 69
3. NOT 70
4. AND 70
5. OR 70
6. ">", "<", "C", and "=" operators 71

V. TROUBLESHOOTING 71

A. The changes you made to the checklist template don't appear in the checklist **74**

B. "Page names must be less than 64 characters" **74**

C. "Unrecognized Command" **74**

D. "IF without ENDIF found in checklist template" **75**

E. "Number expected in LONGTEXT" **75**

F. "Unrecognized PAGE command" **75**

G. "Unexpected End of Checklist Template File" **75**

H. "Invalid heading level for outline" **75**

I. "Unrecognized outline command" **75**

J. "Checklist template line too long" **76**

K. "Ending quote expected" **76**

L. "Second quoted argument expected" **76**

VI. OTHER CHECKLIST TEMPLATE ERRORS 76

A. "No such page exists" **76**

B. "Unmatched quotation marks" **76**

C. "Two conflicting SET's were issued" **77**

D. "Non-existing page reference" **77**

E. "There are no outline nodes specified in the checklist template. WDData requires at least one node on it's outline." 77

VII. OTHER ERRORS AND ERROR MESSAGES 77

A. "The checklist template you have loaded is for the practice system..." 77

B. "No enabled controls are unanswered" 77

C. "Unable to save to DOCS OPEN" 77

D. "Unable to save blast off file" 77

E. "Practice System is not set, or set to a system without a WDData checklist template." 78

F. "Unable to open WDData checklist template" 78

G. "Unable to create the outline control" 78

H. "Unable to open file" 78

I. "The DOCS OPEN support in WDData is disabled. Cannot load the DOCS OPEN data file." 78

J. "Unable to load the Microsoft System Information program" 78

K. "Out of resources" 78

L. "This EXE file has been tampered with. Please get a fresh copy of WDData." 78

M. "Unable to access DOCS OPEN API. Please load DOCS OPEN." 78

N. "Unable to display control" 78

VIII. CONCLUSION 79

I. OVERVIEW OF AUTOMATING YOUR FORMS 80

A. Background Required 80

B. Steps to Create Your Own WinDraft Practice System 80
1. Create the Skeleton 80
2. Adding Your Language 82

C. Add WinDraft Logic 82
1. Identify Variables 83
2. Add Logic 83
3. Test Your Work 84

D. Prepare for Use 84
1. Test The Output 84
2. Document Your System 84

3. Have Others Test Your System 84

E. Roll It Out **84**

II. ADDENDUM: MANUAL STEPS TO CREATING A PRACTICE SYSTEM 84

I. CREATING CUSTOM BLAST-OFF DIALOGS 87

EASILY CHANGING BLAST-OFF DIALOGS FROM TEXT FILES 87

New Tabbed Interface for Blastoff **87**

General Blastoff's BlastoffTemplate.txt **88**

C. WinDraft helper functions Error! Bookmark not defined.
1. INI File Routines 89
2. General Helper functions 90

WinDraft Model Document Author's Guide

I. Author's Guide Introduction -- Simplicity is the Essence

WinDraft was started in 1993. At that time, all of the "document assembly engines" were very complex, didn't support Microsoft Word well, and required that forms be imported and exported into their "work files," with all sorts of extra barriers created. Similarly, Word's merge, fields, and WordBASIC failed to offer a simple and usable way to do "legal document assembly."

WinDraft's mission was to be a **simple** way to develop practice systems that consisted of:

- client information in a basic checklist
- WinWord documents that contained [variables] and [IF ...] rules to "merge" trusts, corporate agreements and other lengthy documents that have optional text in them.

The first WinDraft Author's Guide was just a few pages long. The heart of WinDraft is still contained in these basic concepts. Most of what you will do is create variables in a checklist and insert variables and IF commands in Word documents.

Now, WinDraft has evolved into a program that remains simple at heart but is very powerful. It can automate even the most complex documents. Some of our improvements include:

- new programming commands to handle the special requirements of lawyers
- commands to create a status dialog with moving status bar and messages
- Windows-based, outline-oriented data entry screens
- integration with document management
- output of timeslips and other data into transaction logs
- custom blast-off screens for each system

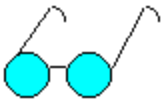
While the following 91 pages may initially appear complex or even overwhelming, we ask you to ignore the "fancy stuff" at first, and focus on what WinDraft was supposed to be in the first place -- a simple system to merge variables and program rules into Microsoft Word documents. Once you "get it," you can move on to the "fancy stuff."

II. Model Document Authors: Who Should Use this Manua

This section is for the lawyer or any other "model document programmer" who wants to create or edit master documents so that others can merge them. It should be read ***after*** reading the User's Guide.

You do not need to read this chapter to merge documents that have been set up by others in your firm or by Eidelman Associates.

In the following pages, you will learn all of the commands needed to create or edit Model Documents that will interact with the information entered at the data entry screen to produce custom client documents. The process of merging documents is described in the User's Guide. **If you are not a system author but are reviewing the text of Model Documents and want to understand the commands in them well enough to know what is going on, we suggest that you read the first parts of this chapter and the one-page command summary found in the inside pocket of the binder.**



Another recommended way to learn the programming commands is to study printouts of the demonstration forms (the ABA Loan Agreement and Babysitting Systems) that are provided with the WinDraft system.

This manual assumes that you know how to do basic editing with Microsoft Word.

III. How to Use This Manual

This manual should be used for programming new WinDraft Model Documents or for altering existing ones.

The WinDraft Model Document Command Summary, which is located in the inside pocket of the binder, contains a quick, one-page reference list of nearly all of the WinDraft "programming" commands. ***If you are in a hurry***, skip this section of the manual and study just the demonstration samples with the command summary in hand.

Programming the WinDraft data entry screen checklists is described in the second half of this tab in a section entitled the WinDraft Checklist Template Author's Guide. Be sure to review that section as well, because the programming of Model Documents and data entry screens is a joint process.

WinDraft also contains a Create New System Wizard that will automatically create a "starter" system that produces the directories, ".INI file" settings, starter checklist templates and model documents for your new system. Using this wizard should be the first thing you do when you want to create a new system.

Persons new to programming should read the next three sections, IV (Concepts and Terminology) through VI (Working with Variables) before attempting to program documents. Experienced programmers should browse the earlier sections and use the Model Document Command Summary and the examples as a programming guide.

This chapter uses the following layout:

?? *[Variable]* Items in brackets, mixed cases and italics are variable names;

?? ***[COMMAND]*** Items in brackets, all capitals, bold and italics are commands;

?? *[;Comment]* Items in brackets and italics preceded by a semicolon are comments.

Variables are those names from the checklist or Model Document which represent document text that varies or may vary each time you generate a document. In a WinDraft model document, a variable represents a merge field where variable information (such as a name, loan amount or contingent legal provision) is processed when the Document is merged.

IV. Concepts and Terminology

WHAT IS A MODEL DOCUMENT?

A Model Document can be thought of as an electronic legal form. In WinDraft, a Model Document is a Word document which contains:

- The plain text of the legal document you will be generating and any Microsoft Word features you want to include such as formatting, automatic paragraph numbering, *etc*., and
- Special variables and commands. All WinDraft variables and commands in a Model Document must be embedded in brackets: `[Variable]` or `[COMMAND]`. As mentioned, variables can represent text that changes with the document or variables can be set up to control the insertion of legal clauses based on the user's entries at the data entry screen.

Your Model Documents should be stored on the same directory of your hard disk or network server, usually in a directory with the name \WinDraft\Sysname (where "Sysname" is your practice system's name).

WHAT WINDRAFT DOES WITH A MODEL DOCUMENT

WinDraft "reads" a Model Document and creates a new Word document from the model by:

- Leaving plain text and formatting codes exactly as they are in the model, thereby supporting all of the word processor's functions and abilities.
- Acting on special text in brackets by (1) filling in `[Variables]` from the data entry screen (these are the answers you supplied in the checklist), and (2) acting on WinDraft `[COMMANDS]`. After WinDraft has completed its "magic," a customized client document is created. The final WinDraft created document can be treated exactly like any other Word document – i.e. edited, printed, saved, *etc*.

V. Basic Command Structure

This section includes some important basic formatting information. Greater detail about commands and formatting will be provided later in this manual.

"[" AND "]" BRACES

All WinDraft Model Document commands and variables must be completely enclosed by a set of brackets. WinDraft operates by scanning the Microsoft Word Model Document for variables and commands (those located within brackets) to know which text to act on.

[IF *RequireAmendmentToBeInWriting* = "Y"**]**
Any future amendment or modification to this Agreement must be made in writing and signed by all parties.
[ENDIF]

In the above example, WinDraft scans the document, finds the brackets, reads the variable, and then checks it against the answer that was given at the data entry screen. If the user answered "yes" to this question, then this paragraph is inserted into the final document. If the user answered "no," then this paragraph is not inserted. Commands are always surrounded by brackets and are in all capitals.

SPACES, TABS, AND RETURNS WITHIN BRACES ARE IGNORED

WinDraft ignores all formatting that occurs **within** brackets. Therefore, spaces (and tabs, indents, and returns) within brackets are optional. This feature, which is discussed in greater detail in "Using Brackets to Enhance Visual Flow," is used to line up the legal logic in your Model Documents. Lining up your programming terms will greatly increase the readability of your logic and enable you to work with and edit complex documents with far greater ease.

[COMMAND] is the same as *[COMMAND ...],*

and *[Variable]* is the same as *[Variable].*

[
COMMAND
]
is the same as *[COMMAND].*

VI. Working With Variables

This section discusses the basics of merging variables into the text of a document. Other programming commands, such as the optional inclusion of text and other special functions, are described in later sections.

A. What Is A Variable?

A variable is a piece of text from the checklist or Model Document that represents information that changes or may change each time you generate a document. WinDraft variables represent merge fields where variable information (such as a name, loan amount, or contingent legal provision) is processed when the document is merged. Variables have the following characteristics:

?? They have names. Each name must be only one word, with no spaces or punctuation, other than a period or underline, in it. The name may contain numbers, but *must* begin a letter (A-Z). However, variable names have no limit to their length. This feature should be utilized to name a variable in as descriptive a manner as possible. Capitals can begin each word, but no spaces can appear in between the words, e.g. *LikeThisExample* or *StateOfIncorporation;*
?? When used in text, variables are surrounded by brackets, [*LikeThisExample*];
?? Periods and underlines can be used in variable names, as in [*Borrower.Name*] or [*Borrower_Name*]. No other punctuation symbols may be used.

Model document:

ARTICLES OF INCORPORATION

These Articles of Incorporation are signed by the **[***Incorp***]** for the purpose of forming a profit corporation pursuant to the provisions of Act 284, Public Acts of 1972, as amended, as follows:

ARTICLE I.

The name of the corporation is: **[***CorporationName***]**

1

Merged document:

ARTICLES OF INCORPORATION

These Articles of Incorporation are signed by the **incorporators** for the purpose of forming a profit corporation pursuant to the provisions of Act 284, Public Acts of 1972, as amended, as follows:

ARTICLE I.

The name of the corporation is: **ACME COMPUTER SUPPLY CO.**

2

As shown in the example above, when WinDraft merges the client data with the Model Document, the variables are replaced with the client-specific information from the data entry screen.

You may use a variable throughout a document and as often as you wish.

VII. Creating New Systems

With time and your knowledge, WinDraft can be used to create expert systems in your various practice systems. The "computer" or "technical" parts of generating a new system (creating the directories for the new system, changing the ".INI" file, *etc.*) are all handled by WinDraft.

A. To Create a New System: Using the Create New System Wizard

If WinDraft is not loaded, load it by pressing <Ctrl-Alt-W> or clicking on the WinDraft button on the toolbar.

Choose Tools | Macro | Macros from the pull-down menu. When the Macros dialog box appears, select CreateNewSystem by highlighting it. Now, choose Run from your list of dialog box options. WinDraft will ask you for the short name of your system (this should be 8 characters or less) and a long descriptive name. WinDraft creates a new directory for this system's documents using the short name, under the main WinDraft directory. WinDraft then automatically creates a sample Model Document for you and places it in the newly created directory. WinDraft also creates a new checklist template for data entry and places it in the template directory. Both the new Model Document and new checklist template take the filenames you specified in the short name field. The long name is the name of your practice system and will automatically be added to the Change Practice System icon list. By following these few simple steps, the technical parts of setting up a new system are avoided. At the end of this process, you will have the basics of your Model Document set up as well as your basic data entry screen program, and an updated ".INI" file.

To learn how to program your Model Documents read the Model Document Author's Guide*;* to write the data entry screens read the Checklist Author's Guide.

VIII. Creating a Model Document

LOAD A FORM OR START FROM SCRATCH

The Create New System wizard will automatically set up your Model Document in the appropriate folder and load the WinDraft style template, which has WinDraft headings and body text formatting built in to sharpen the appearance of your documents. The next steps are to either:

a) type in the document language, the WinDraft commands, the variables, etc. from scratch or from written forms, or

b) alternatively, an existing Word document can be cut and pasted into the Model Document as usual; this document would probably be one of the forms from the firm's form book or a finished general client document. This is the recommended method because it saves the time of re-typing and re-drafting stock phrases. The loaded document needs to be edited – variables must be added for text that potentially changes for each client, and optional text must also be identified.

HOW TO CHANGE PLAIN TEXT

The language or formatting of a Model Document can be edited, saved, and manipulated in the same way as any Word document. No special WinDraft steps or skills are needed. Simply:

- use Word to edit the model,
- change the text to fit your needs, and
- save the document with the same name.

As you edit, be sure not to delete or change any of the *[COMMANDS]* or *[Variables]* or delete any of the brackets.

A. Variables

ADDING VARIABLES TO THE MODEL DOCUMENT

After typing in your general form or loading a form from your hard drive, the next step is to locate document text that varies with the client or situation and replace it with WinDraft variables. Simply give the variable a clear name, place it in brackets and type over the old document text. The variable name, in brackets, must exactly match the variable name that will later be used during checklist template (data entry screen) programming. To illustrate, an old document might read:

?? This Loan Agreement is entered into by John Smith and Jane Taylor on the 10th day of April, 1995. It will be interpreted under the laws of the State of Michigan.

The "programmed" Model Document will read:

?? This Loan Agreement is entered into by [*Lender*] and [*Borrower*] on the [*DateOfAgreement*]. It will be interpreted under the laws of the State of [*ChoiceOfLaw*].

NAMING VARIABLES: USE LONG VARIABLE NAMES FOR CLARITY

When naming variables, it is very helpful to choose names that are descriptive. For example, if you are creating a variable to represent the number of directors of a corporation, use a name like [*NumberOfDirectors*]. Since spaces cannot be used, separate the words in the variable name by using capital letters:

[*NumberOfDirectors*]

Again, note that there are no spaces between words. For ease of reading, a capital letter is used to signal the beginning of each new word.

After you have programmed the model document, you will use these exact same variable names to program the checklist template---the data entry program that asks the user all the various client and legal questions that are needed to draft the relevant document or set of documents. Checklist Template programming is very similar to Model Document programming and is explained in the *Checklist Template Programming Guide*.

PERSONAL PRONOUNS

In naming personal pronoun variables, use names that make it easy to understand the Model Document:

```
[HeShe] may act in [HisHer] discretion ...
as seems best to [HimHer].
```

Personal pronouns and verb tenses can also be handled using Word fields. For examples, see the Model Document's for the Babysitting and Estate Planning demonstration programs.

B. SET

SET

One way to tell WinDraft the value of a variable is to **SET** it by using the [**SET**] command. Thus, in the above example, a law firm in Michigan may want to have all agreements interpreted under Michigan law, thus the variable [*ChoiceOfLaw*] can be set to equal Michigan. Whenever this variable

[*ChoiceOfLaw*] appears in the text, Michigan will be inserted in its place. This makes it unnecessary to ask this question in the data entry screen and avoids wasting the user's time in asking a question whose answer is unlikely to change. If, however, the variable ever did need to be changed to meet a client's wishes, the command could found in the Model Document and edited to reflect the appropriate change:

The above example would appear in a Model Document as:

?? [**SET** *ChoiceOfLaw* = "Michigan"]
This Loan Agreement is entered into by [*Lender*] and [*Borrower*] on the [*DateOfAgreement*]. It will be interpreted under the laws of the State of [*ChoiceOfLaw*].

Notice that the data to the right of the equal sign in the **SET** command is not surrounded by brackets, but rather by quotation marks. When you assign a variable an answer it must appear in quotation marks. (In computer jargon, the [**SET**] command is called an "assignment statement" because it assigns the value of the variable to the variable.)

These variables will be in the memory of the computer only when you are merging the specific document or related documents which contain the [**SET**] command.

The **SET** command is useful also with gender terms. See the Babysitting example, and in particular the file BSGender.DOC.

C. ASK

ASK

The **ASK** command tells WinDraft to ask the user a question during the process of merging a Model Document with a data file. This allows prompting for information without checklist screens.

The text within the set of brackets that follows the **ASK** phrase, [*Variable*] will add the variable to the document text. From that point forward, insert the second variable wherever the variable is to be added to the text.

?? [**ASK** *Variable* "question to ask user"] [*variable*]

This command,

?? [**ASK** *Incorporator* "What is the name of the incorporator?"]

[*Incorporator*]

causes WinDraft to prompt the user for the incorporator's name as the document is being merged. The variable [*Incorporator*] can then be used anywhere in the document, but it will not be retained in the data file after the document has been merged.

Since the information entered in an ASK command is not saved, once you have learned your way around WinDraft a little, you'll want to use the checklist screens instead. ASK commands can be a quick way to get started programming model documents, though.

IX. Beyond Merging Variables: Learning To Program Documents

A. [;; Comments -- The more the Better]

The computer will completely ignore all commands and text between brackets that begin with a semi-colon.

?? [; This is a comment.]

?? [; This is also a comment. Notice that comments can be as long as you want and can be spread over several lines, as long as they are placed between the brackets and preceded with a semi-colon.]

As your document becomes more complex, with many nested **IF** statements for example, it will also become more difficult for others (and even for you) to follow. Good programming practice ***requires*** that you use comments liberally. ***Do*** take the time to add explanatory notes to your document as you go. Its logic may seem obvious to you as you program, but will be confusing when you or someone else comes back to it months later.

Comments should also include:

?? legal references, if a paragraph was included because of a particular IRS or court ruling or a statute, and

?? information about who made changes to the forms and the dates of such changes.

B. Basic Programming Concepts

In earlier days of programming, a typical computer program would look like this:

```
100  IF A1$ > B1$ GOTO 400
110  ELSE GOTO 500 ...
```

A more modern concept called "structured programming" is now used. Programs should be as easy to read and understand as possible. It should be

possible to use "top-down design," which means organizing the document from the most general to the most specific. Programming can be laid out visually in such a way that the reader can understand the flow of logic by looking at the level of indenting from the left margin.

BALANCING ENDING COMMANDS

Many programming commands have a complementary closing command so that the computer will know the limits of what text and commands to act on when it executes the command. The closing commands always begin with the word "END," as in:

IF... ***ENDIF***

If your Model Documents are not working correctly, start by looking for missing or misplaced ENDIFs. When you merge a document, WinDraft will also prompt you with an error if there is either a missing **IF** or **ENDIF** command.

The best way to ensure that you always include a matching **ENDIF** statement is to insert the matching **ENDIF** at the same time you start typing the command. For example, whenever you use an **IF** command, type in both [**IF**] and [**ENDIF**] first, then fill in the middle. This way you won't forget to add the **ENDIF**.

WinDraft allows you to nest command structures as deeply as you want, as long as you make sure that the **ENDIF** statements are in the right place. Complete beginning and ending statements must be nested within each other. This gives you the power to build and automate very complex documents

INDENTING NESTED COMMANDS

When the commands become complex, it is essential to format the documents so that each beginning command visually lines up with its matching end command. This can be done by indenting appropriately.

?? **[IF** *SecurityDeposit* **= "Y"]**
This is text that is included in the document only if the first "IF" is true. **[**
IF *EarningInterestOnSecurityDeposit* **= "Y"**
] This is text that is included only if both IF statements are true. **[**
ENDIF] [; This closes the inner IF clause]
[ENDIF] [; This closes the outer IF clause]

By using this technique, it is fairly easy for the document programmer to understand the flow of the logic.

(Note: When commands and model text are interspersed, as in the previous example, it is easy to end up with tabs, spaces or hard returns where you don't want them. To avoid this, see the section entitled B. Proper Placement of Brackets on page 41.)

WinDraft completely disregards all formatting (tabs, returns and spaces, as well as italicized, bold or underlined text) within brackets. In addition, it is important to realize that MS Word uses the paragraph formatting that is applied to the end of the paragraph as the formatting for the entire paragraph, as explained in the section entitled B. MS Word Paragraph Attributes and Hard Returns on page 37, below.

C. IF Statements

WinDraft lets you teach your documents to "think like a lawyer." When a lawyer drafts a document, he or she must think through a number of RULES:

IF the corporation is a Subchapter S corporation,
THEN insert a resolution selecting Subchapter S.

IF there are two or more classes of stock,
THEN Subchapter S is not a valid election.

IF there is only one director,
THEN put in just one signature line,
AND say "sole director"
ELSE
put in more than one signature line,
AND say "all of the directors."

The most important way that WinDraft saves time and increases the quality of your work is by considering rules as it assembles documents. You have complete control over the rules.

"**IF** statements" work in the following manner:

Model Document	*Merged Document*
[**IF** *SubchapterS* = "Y"] BE IT RESOLVED, that the corporation hereby elects to be treated as a small business under the provisions of the Internal Revenue Code, as amended, and to be taxed under the provisions of Subchapter S. [**ENDIF**]	BE IT RESOLVED, that the corporation hereby elects to be treated as a small business under the provisions of the Internal Revenue Code, as amended, and to be taxed under the provisions of Subchapter S.

MARKING OPTIONAL CLAUSES AND PROVISIONS: THE IF STATEMENT

Optional clauses and paragraphs are included in the final document only if the user has "chosen" or "selected" them at the data entry screen level. An **IF** statement works by checking whether its condition has been met. If the statement is true or the variable is equal to the variable selected in the data entry screen, the text and commands between the **IF** and **ENDIF** will be included. If the statement is not true, then the text and commands between the **IF** and **ENDIF** will be ignored and skipped over.

In the case below, there is a data entry screen question that asks for a yes or no response as to whether a sub-chapter S corporation is elected. If the user answers yes, the following provision will be included in the merged document, but if the question is answered "No," the provision will be skipped over.

[**IF** *SubchapterS* = "Y"]
    Be it resolved that the company elects to be taxed as a sub-chapter S corporation.
[**ENDIF**]

Another common use of the **IF** statement is to execute commands which set personal pronouns, subject-verb agreement, or other variables:

[**IF** *DirectorsGender* = "M"]
    [**SET** *DirHisHer* = "his"]
    [**SET** *DirHimHer* = "him"]
    [**SET** *DirHeShe* = "he"]
[**ENDIF**]

An **IF** command can also be used to check a response to a data entry screen question such as Radio button or Code button answer. These questions were described in the *User's Guide*. The inclusion of a paragraph may depend on the user's answer to those questions. In that case, the **IF** statement is set up

as follows:

?? [**IF** *TheAnswerGivenAtTheDataEntryScreen* = "A particular answer"]
If the answer to the question on the data entry screen is the particular answer above, then this paragraph will be inserted. If the answer does not match, then this paragraph will be skipped over.
[**ENDIF**]

?? [**IF** *BorrowerType* = "Corporation"]
The corporation is organized under the laws of the state of [*StateOfIncorporation*.].
[**ENDIF**]

In the second example, there would have been a data entry screen question that asked whether the borrower was a corporation, partnership or individual. The variable name attached to that question would have been *BorrowerType*, *e.g*., the following command would have been in the programming for the data entry screens (See WinDraft Checklist Author's Guide, p.50 et seq.).

CHOICE *BorrowerType* "Is the client a"
"Corporation" "Corporation"
"Individual" "Individual"
"Partnership" "Partnership"
END CHOICE

If the borrower had been an individual, he would not be asked a question about his state of incorporation at the data entry level, and this paragraph would be irrelevant to him, so it is left out of the final document.

ELSE -- THE ALTERNATE COMMANDS

The [**ELSE**] command goes between an [**IF**] and [**ENDIF**] statement when there is alternate text or other commands to be executed when the [**IF**] statement is not satisfied.

For example, in a corporate document, the two optional, alternate paragraphs for selecting preemptive shareholder rights might be programmed in a Model Document like this:

[**IF** *PreemptiveRight* = "Y"]
The shareholders shall have a preemptive right to purchase additional stock to preserve their pro rata ....
[**ELSE**]
The preemptive right of shareholders to purchase additional stock is denied.

[**ENDIF**]

Another example is,

[**IF** *CumulativeVoting* = "Y"]
    Elections of directors shall be conducted on a cumulative voting basis so that any shareholder may vote all of his or her shares for one director
[**ELSE**]
    There shall be no cumulative voting.
[**ENDIF**]

ELSEIF

Use the [**ELSEIF**] command if there are more than two options, and only one can be correct. The first option should follow an [**IF**] command. The [**ELSE**] statement can also be used in conjunction with the **ELSEIF**.

[**IF** *CorpBookSupplier* = "Blumberg"]
    Julius Blumberg, Inc. ....
[**ELSEIF** *CorpBookSupplier* = "Acme"]
    Acme Book and Seal, Inc. ...
[**ELSEIF** *CorpBookSupplier* = "LPI"]
    Lawyer's Publishing, Inc. ...
[**ELSE**]

[**ASK** *CorpBookSupplierName* "What is the name of the corporate minute book vendor?"]
[*CorpBookSupplierName*] ...
[**ENDIF**]

Notice that the **IF** conditions will be processed in order. When the condition is satisfied with a true statement, the computer will jump to the **ENDIF** and will not consider the other alternatives, even if other alternatives below the first true one are also true.

For maximum speed, put the condition that will usually be true first, so that the computer does not always have to consider conditions that will usually be false.

D. INCLUDE

INCLUDE

We strongly suggest that you use the **INCLUDE** command to break down complex documents into simpler modules and to hide complex programming. The **INCLUDE** command can also be used to store subprograms that are

called into use by many different Model Documents or practice systems. Instead of re-programming the logic and text for each Practice System or Model Document, set the frequently used Model Document out on its own, so that it can be called as needed with an **INCLUDE** command. The logic for setting gender terms or signature lines is also a good candidate for an **INCLUDE** and its sub-file.

WHAT INCLUDE DOES

The ***[INCLUDE "filename"]*** command tells WinDraft to include the text and commands of the sub-file named in the command at that point in the Model Document.

DOCUMENT WITH INCLUDE COMMAND	**SUBROUTINE FILE CALLED BY INCLUDE**
[; Sample file with **INCLUDE** command] This is text before the INCLUDE command. The text from the other file will be appear after this paragraph. ***[INCLUDE "subdoc1.doc"]*** This text will appear in the output document after the text from the subfile is inserted in this document. The value of the variable set in the subfile is: [city]. Variables can be created in subroutines, as in this example.	[; This is a separate subroutine document called SUBDOC1.DOC.] ***[SET City = "St. Louis"*** ***]*** This text will be included in the Model Document that has the INCLUDE command in it, just as if the text of this document were in the original before it is merged.

MERGED OUTPUT DOCUMENT

This is text before the INCLUDE command. The text from the other file will appear after this paragraph.

This text will be included in the Model Document that has the INCLUDE command in it, just as if the text of this document were in the original before it is merged.

This text will appear in the output document after the text from the subfile is inserted in this document. The value of the variable set in the subfile is: St. Louis. Variables can be created in subroutines, as in this example.

SYNTAX OF THE INCLUDE COMMAND

To use the include command, you:

?? type **INCLUDE** in all caps, and surround it, as well as the program to be called up, in brackets: [**INCLUDE** "SubChapt.doc"]

?? surround the name of the sub-document to be merged or called in quotation marks.

Note: If you just put in the name of the file to be included, without the full path name, WinDraft will look on the "subdocs" directory as set in WinDraft.INI. If you want WinDraft to look elsewhere, put in the full path name, such as: "C:\Dirname\Subdirname\FileName.DOC."

Note also that the **INCLUDE** command will append a paragraph return after any text that has been **INCLUDE**d (this can be seen in the example above).

SUB-DOCUMENT NAME STORED IN VARIABLE

You can also store the file name, according to the instructions above, in a variable name and use the variable name in the include command:

[**SET** *filename* = "2classes.doc"]
[**INCLUDE** *filename*]

This is particularly helpful when using programming functions to **INCLUDE** different sub-documents under different circumstances. For example, if you have a different closing paragraph for each lawyer in the firm stored under the lawyer's initials JAE.SUB, ABC.SUB, etc., a command can calculate the file name using the **[SET]** command, then **INCLUDE** using the variable name. (Examples of how to calculate a variable name using the "+" are provided under the Operators section in this manual on page 31.)

STORE LONG DOCUMENT MODULES IN SUB-DOCUMENT

[**INCLUDE**] command can also break down long documents into modules. This is particularly helpful when you have large amounts of text embedded between [**IF**] ... [**ENDIF**] commands.

For example, if when you have two classes of stock, you use several pages of paragraphs stating limitations on voting, preferences, *etc*., it is much easier to understand the master Model Document if you put the long special provisions in and hide complex programming in a separate document. The document can be called 2CLASSES.DOC and use the **INCLUDE** command:

[**IF** *NumberOfClasses* = "1"]
    There shall be one class of common stock, of par value $[ParValue], with no preferences or limitations.
[**ELSE**]
    [**INCLUDE** "2classes.doc"]
[**ENDIF**]

This makes it much easier for the document programmer to follow the flow of the document and the nesting of [**IF**] statements

HIDE COMPLEX PROGRAM-MING IN SUB-ROUTINES

Your goal is to have a Model Document which looks like a legal document, not a computer program. One way to do this is to hide complex programming in a separate document that can be included. The separate document can be called via a subroutine.

A second benefit is that the subroutine can be used over and over again in different Model documents, so that the document programmer does not need to reinvent the logic.

An example is a subroutine to construct a person's name. Suppose the data for people's names is stored in a database in six fields, called first, middle, last; Mr./Ms.; suffix (for Jr., III, etc.), and lawyer (yes/no) for "Esq." (for lawyers' names in correspondence). You must use a series of **IF** statements and **[SET]** commands to build the full name for use in letters and signature lines. These commands include lines that tell the computer, for example, to include the middle name or initial and a space only if the middle name field is not blank.

You can put the commands to create the [*DirectorFullName*] variable in a sub-routine document called DIRNAME.DOC. Then, every time you need a signature line for directors, enter the following:

Signed this ___ day of June, 1996.

[**INCLUDE** "DIRNAME.DOC"]

_____
[*DirectorFullName*]

This minimizes the number of commands required in a Model Document, making it easier to follow the logic of the legal document, especially for the "computer-impaired," and saves time by avoiding the need to re-program this logic with subsequent practice systems.

E. RULES

RULES

The **RULES** command reads and executes WinDraft commands from a separate text file. It is similar in function and use to the **INCLUDE** command but it is used to optimize the speed of WinDraft when merging documents. Like **INCLUDE**, it calls in another subdocument. It should be used in sections of the document which are all logic, *i.e*. WinDraft programming commands, with no text

RULES differs from **INCLUDE** in the following respects:

- The subfile called by **RULES** is ASCII text instead of a Word document.
- The subfile contains only WinDraft commands, one to a line. The commands cannot be split across lines.
- The commands in a **RULES** subfile are processed about 10 times faster.
- Brackets around commands are optional in the **RULES** subfile.
- The subfile called by **RULES** may not itself call another file (i.e. may not contain a **RULES** or **INCLUDE** command).

Example:
?? **[RULES "GENDER.RUL"]**

X. Other Commands

The following summarizes some of the additional commands not fully described in the preceding sections:

A. GETDATA

GETDATA

?? ***[GETDATA]***
?? ***[GETDATA "****filename****"]***

Almost every model document needs to access client information from the WDData data entry module. This command that accomplishes this is **GETDATA**, which imports data from the most recent data entry file into Word. The **GETDATA** command would normally be invoked at the beginning of the top-level Model Document. It need not appear more than once, nor does it need to appear in subdocuments. **GETDATA** can also be used with a filename, for example
?? ***[GETDATA*** *"LenderData.txt"*
This allows a model document to access data from more than one source. In the case above, for example, information about the lender might change less often than borrower data. It might make sense, therefore, to leave lender questions out of the main loan data entry template and put them into their own checklist template instead.

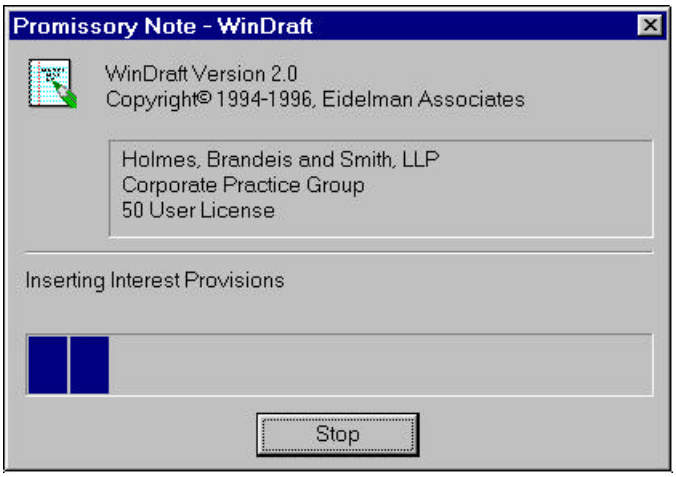
B. DOCTITLE

DOCTITLE

?? ***[DOCTITLE "****Text****"]***
?? ***[DOCTITLE*** *"Promissory Note"****]***

This command is used to tell the user which document WinDraft is merging on the screen. It should be placed at the beginning of the document and can be helpful when a user chooses to create a long list of documents and wishes to see which one is currently being produced.

The example above created the heading on the dialog box shown below.



C. PROGRESS

PROGRESS

?? ***PROGRESS*** *"number"]*
?? *[**PROGRESS** "11"]*

The **PROGRESS** command updates the thermometer bar in the WinDraft dialog box. This should be followed by a numeric constant. In the example, ***[PROGRESS "11"]*** indicates that the document merging process is 11% complete. This command is optional. It should be placed at the appropriate point in the Model Document to show how far along WinDraft is in merging the document. Be sure that the numbers in the PROGRESS statements increase as you progress through the model documents. Progress bars which run backwards are disconcerting, to say the least.

D. STATUS

STATUS

?? *[**STATUS** "Text"]*
?? *[**STATUS** "Inserting Interest Provisions"]*

This command tells WinDraft to insert the quoted text in the WinDraft dialog box as the document is being merged (see the illustration above). This command informs the user which provisions are being inserted as WinDraft creates the custom document. The text must be enclosed in quotation marks (or be a more complex WinDraft expression).

E. SYSLOG

SYSLOG

?? *[**SYSLOG** "Text"]*
?? *[**SYSLOG** "Drafting bill for " + ClientName]*

This command outputs information into WinDraft's log file. The log file is determined by the ".INI" file's LogFile setting; please refer to the ".INI" file documentation for more information. The example above would output a line which might read

```
Drafting bill for ABC Corporation
```

in the log file.

F. LOG

LOG

?? *[**LOG** "LogFileName", "Text"]*
?? *[**LOG** "Timeslip.log", "JAE,12/2/95,JonesCorp,0029,2.5,Drafted asset purchase agreement"]*

This command outputs information into a specified file. **LOG** may be useful in setting up integration with billing systems, mail, and so forth.

G. PROFILE

PROFILE

?? *[**PROFILE** Abstract = "Text"]*
?? *[**PROFILE** Abstract = HusbandName + "revocable trust with generation skipping"]*

The **PROFILE** command is used to export information from WinDraft into other programs, such as document management. For instance, WinDraft's **PROFILE** command can export information to Word's document summary fields or DOCS OPEN profiles for each document it "blasts off". See the User's Guide section called *Working with Automatic Document Summaries*, and the Technical Appendix on DOCS OPEN integration (in the file WDAPPNDX.DOC) for more information..

H. DISPLAY ON/OFF

DISPLAY ON/OFF

?? ***[DISPLAY on/off]***
?? ***[DISPLAY OFF]***

The **DISPLAY** command determines whether the merging of a given part of

the document is displayed on screen. This command only has an effect when the user has Watch Assembly turned on. By default, WinDraft displays all assembly to the screen. However, if WinDraft finds a *[**DISPLAY** OFF]* command, then it will stop displaying any output until it finds a matching *[**DISPLAY** ON]* command. This is useful for hiding complex logic in the document during the merge process.

I. FIELD

FIELD

?? *[**FIELD** "Text"]*
?? *[**FIELD** "DATE * MERGEFORMAT"]*

This command inserts the evaluations of expressions into the document as a Microsoft Word field. The example above inserts a field with the current date into the document. For more information about the power of Word fields, see your Word documentation or, from the Word menus, select Insert|Field and mouse around.

Note that the **FIELD** command is used to insert Word fields into a document. The **REF** function is used to get Word field variables back into WinDraft variables. See **REF** under the Functions section (the next section) for more information.

J. HALT

HALT

?? *[**HALT**]*

Stops WinDraft. This is primarily for debugging purposes, although it could be used as part of an error checking system to end a document prematurely if data is not entered correctly.

K. FOR

FOR…
DO

?? *[**FOR** I = 1 TO NumberOfChildren INCLUDE "Children.doc"]*
?? *[**FOR** I = 1 TO 10 DO SET Name(I) = FirstName(I) + " " + LastName(I)]*

The FOR command is used to do something repeatedly. It can be matched with the FOR command in the checklist template (as in the first example about, where *NumberOfChildren* is the index variable in a FOR command on the checklist).

If there is only one command you need to perform repeatedly, you may write something of the form [**FOR** I = 1 **TO** x **DO** command], where command is the command you want executed repeatedly. Otherwise, you must put the commands to be executed in an INCLUDE or RULES file, and follow the syntax [**FOR** I = 1 **TO** x **INCLUDE** file] or [**FOR** I = 1 **TO** x **RULES** file].

Optionally, you can proceed by steps of something other than 1. For example [**FOR** I = 1 **TO** 10 **STEP** 2 ...] will count I=1, 3, 5, 7, 9, and perform commands you specify at each I.

J. GETDB

GETDB

?? *[**GETDB** PersonalInfo(), **FILE** "db2.mdb", **FROM** "people", **GET** "LastName, FirstName, Age, CLMat"]*
?? *[**GETDB** Author, **DSN** "ClientMatter", **FROM** "blEmployee", **GET** "LastName, FirstName", **WHERE** "Email='" + AuthorLogin + "'"]*
?? *[**GETDB** PersonalInfo() **FILE** "db2.mdb" **SQL** "SELECT * FROM People"]*
?? *[**GETDB** PersonalInfo() "DSN=People", "SELECT * FROM People"]*

Reads information from an ODBC compliant database. It starts with the keyword **GETDB** and a variable name. If the given variable name ends in an open/closed parentheses pair (), then the information is read into an array. Otherwise, it is read into one variable, and WinDraft gives an error if more than one record is returned by the query.

Next is specified where to read the data from. This can be done in three ways. The first is to use the **FILE** parameter to read from a hardcoded file (with the default file type being Microsoft Access, the **DRIVER** modifier can be used to select any ODBC compliant driver for the file). The second and more general way is to use the **DSN** parameter to give a general ODBC DSN. See the ODBC documentation for more information on DSNs. The third, and most general way, is to simply supply an ODBC connect string (such as "DSN=WDPeople") as the second parameter of GETDB, without any other indicators.

Finally, you specify what should be read. Again, there are two ways to do this. The first is to have a **GET** parameter with a list of the desired fields, together with a FROM parameter to give the desired table, and optionally a **WHERE** parameter which limits the choice of records. The second is to have a **SQL** parameter with a general **SQL** command (the SQL indicator can also be omitted, as in the last example above). See the SQL documentation

for more information on the SQL language, or use a tool like MS Query or Access to build your query.

The information will be read into *VarName(*).FieldName* or *VarName.FieldName*, depending on whether an array () was specified in the **GETDB** command. For example, in the first **GETDB** command above, we will have
PersonalInfo(1).LastName = "Jones"
PersonalInfo(1).FirstName = "Tom"
PersonalInfo(1).Age = "35"
PersonalInfo(1).CLMat = "1000"

PersonalInfo(2).LastName = "Smith"

And so forth.

XI. Functions

A "function" is a special kind of modifier that is followed by parentheses and modifies the text, number or expression found in the parentheses. The following functions exist within WinDraft.

A. UC, LC, IC and PC to Capitalize Letters

UPPER CASE

?? ***[UC(Variable)]***
?? ***[UC(PlaintiffsName)]***
??
The **UC** function converts the specified variable or expression to upper case. In the example above, this modifier causes WinDraft to print out the Plaintiff's name in all capital letters in the merged document. Thus if, in response to a question at the data entry screen asking for the plaintiff's name, the name "john smith" was typed in, it will appear in the Document as JOHN SMITH.

LOWER CASE

?? ***[LC(Variable)]***
?? ***[LC(PlaintiffsName)]***
??
The **LC** function converts the specified variable or expression to lower case. The example will cause the plaintiff's name to appear in the merged document as john smith.

INITIAL CASE

?? ***[IC(Variable)]***
?? ***[IC(PlaintiffsName)]***

The **IC** function capitalizes the first letter of a sentence, and leaves the rest alone. The example would come into the document after **IC** as John smith.

PROPER CASE

?? ***[PC(Variable)]***
?? ***[PC(PlaintiffsName)]***

The **PC** function capitalizes the first letter of each word and converts the rest to lower case. The example will cause the plaintiff's name to appear in the merged document as John Smith.

B. LEFT and RIGHT to break up strings

LEFT

?? ***[LEFT(State, 1)]***

The **LEFT** function returns the first several letters of the input expression. For example, **LEFT**("Michigan", 1) would return "M", **LEFT**("Michigan", 2) would return "Mi", and so forth.

RIGHT

?? ***[RIGHT(State, 1)]***

The **RIGHT** function returns the last several letters of the input expression. For example, **RIGHT**("100,000", 1) would return "0".

C. REF to Read a Word BookmarkField

REFERENCE FIELD

?? ***[REF("Field")]***
?? ***[SET GrantorHeShe = REF("GrantorHeShe")]***

The **REF** function looks up an expression given as a Word bookmark and returns the text of the bookmark. If there was a Microsoft field command in the Model Document which sets the `GrantorHeShe` field to something, one could transform this into a WinDraft variable of the same name by the example above.

D. MERGEFIELD to Read a Word Mail Merge Field

MAIL MERGE

?? ***[MERGEFIELD("Field")]***
?? ***[SET** LastName = **MERGEFIELD("LastName")]***

FIELD

The **MERGEFIELD** function looks up an expression given as a Word mailmerge field, and returns the value of the field. This might be used when running WinDraft on a document which was mail merged with a contact management database--

E. SPELLOUT to Spell Numbers

SPELLOUT

?? *[**SPELLOUT**(NumericVariable)]*
?? *[**SPELLOUT**(LoanAmount)]*

SPELLOUT converts a numeric variable to a spelled-out version of the same thing. **SPELLOUT** supports numbers as high as 4,294,967,295. For example, if at the data screen the user inputs 500,000 in response to a question asking for the amount of a loan, WinDraft will spell this amount out in the Model Document—five hundred thousand will appear in the merged Document.

SPELLOUT ignores leading dollar signs ($), as well as any commas that may appear in the number.

F. Math functions

ADD
SUBTRACT
MULTIPLY
DIVIDE

?? *[**ADD**(2, 2)]*
?? *[**SUBTRACT**("3", "1")]*

While WinDraft is designed for text processing and not number crunching, it does support some elementary arithmetic functions. These functions all take 2 arguments, convert any string (or variable) arguments to numbers by looking at all the digits up to the first non-numeric character, and perform the appropriate operation on them. For example, *[**SUBTRACT**("3", "1")]* will give 2.

INC
DEC

?? ***[INC(Counter)]***
?? ***[DEC(Counter)]***

INC(Counter) (short for increment) is equivalent to **ADD**(Counter, 1). **DEC**(Counter) (short for decrement) is equivalent to **SUBTRACT**(Counter, 1). They are both often useful when working with **FOR** loops.

XII. Operators

The following "operators" can be used in logical expressions within **IF** clauses.

NOT, **AND**, **OR**

These are the logical connectives to allow for compound statements and negation. The order of precedence is **NOT**, **AND**, **OR**.

A. NOT

THE "NOT" OPERATOR

The **NOT** operator allows a paragraph, clause or question to be inserted or asked if a condition is *not* met. An example of the use of **NOT** is:

?? *[**IF NOT** TrusteeType = "Corporation"]*
[; The trustee is an individual]
...
*[**ENDIF**]*

B. AND

THE "AND" OPERATOR

The **AND** operator allows a paragraph, clause or question to be inserted or asked if all of several conditions are met. An example of the use of **AND** is:

?? *[**IF** CaseType = "Auto Negligence" AND PersonalInjuries = "Y"]*
PIP benefits...
*[**ENDIF**]*

C. OR

THE "OR" OPERATOR

The **OR** operator is to allow a paragraph, clause or question to be inserted or asked if one of several conditions is met. An example of the use of **OR** is:

?? *[**IF** EntityType = "Corporation" **OR** EntityType= "LimitedLiabilityCompany"]*
The entity is organized under the laws of [StateOfIncorporation].
*[**ENDIF**]*

When using the **OR** command, the variable name that is being checked must be retyped each time, like *EntityType* in the example above.

D. = and other Comparison Operators

THE ">", "<", "=" and "C" OPERATORS

These are the four comparison operators:

>	greater than
<	less than
=	equals
C	contains

For example:

?? *[**IF** NumberOfInitialDirectors > "3"]*
Missouri law permits the number of directors to be changed in the manner provided in the bylaws.
*[**ELSE**]*
The number of directors must be stated in the Articles of Incorporation.
*[**ENDIF**]*

?? *[**IF** NumberOfInitialTrustees = "1"]*
I appoint the following as successor trustee--[*SuccessorTrustee*].
*[**ENDIF**]*

The **C** operator will be evaluated as true if the expression to the left of the **C** "contains" the expression following the **C** as a substring.

For example, if the person's name is Johnson, the following will both be true:

?? *[**IF** Name **C** "son"] ...*

?? *[**IF** Name **C** "John"]*

For more information about the C operator, see the discussion of "Combo Codes" on page 42 later in this chapter.

E. + to Join Text Strings Together

THE + OPERATOR

?? *[String1 + String2]*

This operator is used to join two strings together. (In computer jargon, this is called "concatenation." The two character strings are "concatenated" together.) For example, you might want to concatenate several variables in the following manner:

?? *[**SET** GrantorAndSpouse = GrantorName + " and "+ SpouseName]*
?? *[HusbandFirstName + " " + HusbandLastName]*

This would combine the contents of the *GrantorName* variable, and the *SpouseName* variable, with the word "and" (which is our "string constant") in between. The spaces around the "and" are required in the above to avoid having the two names run together as one word. In the second example, quotes are needed around a blank space to separate the two names.

F. Including Quotation Marks in String Constants

DUAL QUOTATION MARKS

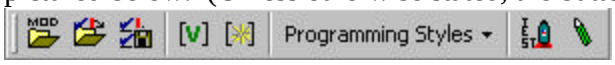
If quotes are needed in a string constant, WinDraft will evaluate two quotation marks next to one another as a single quotation mark, rather than as the end of a constant. This feature is especially useful for the PROFILE command which fills out the DOCS OPEN profile forms.

Functions can also be used in an expression. For example, if the author wanted the names to be uppercase in the above example, he could use

?? *[SET GrantorAndSpouse = UC(GrantorName + " and " + SpouseName)]* or

?? *[SET GrantorAndSpouse = UC(GrantorName) + " and " + UC(SpouseName)]*

XIII. WinDraft Author's Toolbar

We have programmed a number of special tools and macros to assist you in authoring Model Documents. Those that have a button associated with them have the button pictured below. (Unless otherwise stated, the button appears on the Author Tool Bar: 

). If the toolbar is not displayed, you can display it by selecting it from the pull-down menu under Tools|WinDraft Options.

A. The Open Model Documents Button

OPEN MODEL DOCUMENTS

Open Model Documents

This button is similar to the File|Open command or button, except that it will open up the subdirectory where the model documents for the currently selected practice system are located.

B. The Programming Style Buttons

PROGRAM-MING STYLE

Programming Styles popup menu

In order to simplify the reading of WinDraft commands in Model Documents, we have created three styles called "Programming," “Programming 2” and “Programming 3." These are character-format styles for programming text. Initially, we have chosen to base them on the normal font and to italicize them. The text color is blue, red or magenta, respectively, as shown on the menu icons. The different colored styles are useful in complex documents to keep nested IF statements clearly matched with their corresponding ENDIFs. You can change the Programming Styles to make the programming commands appear any way you like, such as in Courier font.

To use one of these buttons, position the cursor between a pair of brackets, click on the Programming Styles toolbar, and select a Programming Style menu item. The brackets and the text within them will automatically be formatted to this “Programming Style.” If your document didn’t contain a style by that name one will be added. To change this style to meet your preferences, choose Format Style, then choose “programming” from the list of styles on the left, click “modify,” then on the new screen choose “format” font and make the changes to match the results you desire (for instance, select the Courier font).

There are also styles designed to be applied to IF statements. IF styles 1, 2, 3, and 4 are by default applied to an entire line of code . The IF and ENDIF menu items apply the style to the IF under the cursor, and search for the next ENDIF to apply the style to automatically.

C. The Blast-Off Test Button

BLAST-OFF TEST ICON

Blast-Off Test

Clicking on this icon pulls up the Blast-Off dialog box with which you can choose the documents you want to merge. Performing this operation is the

same as selecting Blast-Off from the data entry screen but saves you the trouble of having to load or switch to WDDATA, the data entry screen. Blast-Off Test re-uses the most recent set of checklist data. It is useful for testing and troubleshooting since it allows you to modify and test your model document programming repeatedly without having to switch back and forth between your model document and the checklist.

D. Buttons to Work with Checklist Templates for WDData

Icons to load and save checklist templates

These buttons are described in more detail in the section on working with checklist templates. They are used to open the checklist template files for editing and to save checklist template files back to disk.

Open Checklist Template. This button will open the checklist template in Word for editing.

Save Checklist Template
This button saves the file open on the screen in a Word (*.DOC) format for editing by authors and as a text file (*.INT) so that WinDraft's data entry program can read it. As you edit the checklist template, click this button to save your changes (Be sure to close WDData and reload it when you have saved new changes.)

E. The Assemble Document Now on Screen Button

ASSEMBLE DOCUMENT NOW ON SCREEN

Assemble Document Now on Screen

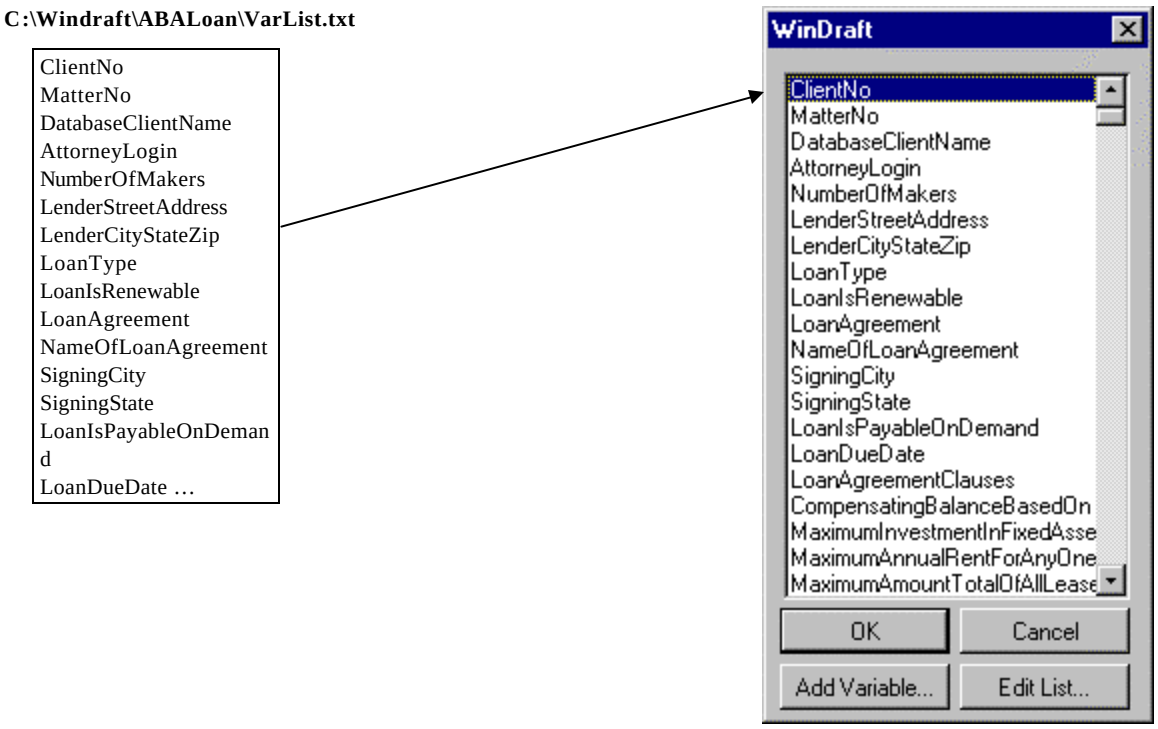
Clicking on this icon causes WinDraft to merge the document you currently have on screen. It functions like the Blast-Off Test Icon, but does not pull up the WinDraft Blast-Off Dialog box. The icon is useful for troubleshooting not only Model Documents, but also subroutine documents (e.g. gender documents) which otherwise could not be assembled and tested for errors.

XIV. Authoring Assistant Macros in Word

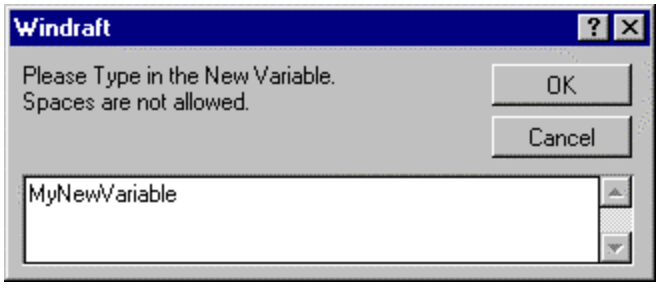
A. Insert Variable

To make model document authoring easier, we have added a button (WDInsert Variable) on the Author toolbar which presents the user with a list of available variables from the current practice system from which he/she can choose to immediately insert into the current document. A list box in the Insert Varible form is filled with variables from a file called VarList.txt stored in

the root directory of the current practice system. Varlist.txt is automatically updated at the end of each assembled document with all variables available to that document. To insert one of the variables simply double click on the variable in the list box or click Ok.



When you press Add Variable…, you are presented with a small dialog box into which you can enter the name of the variable or text string you wish to make available in the Insert Variable form for the current practice system in the future. Then, MyNewVariable will be appended to the end of the VarList.txt file.

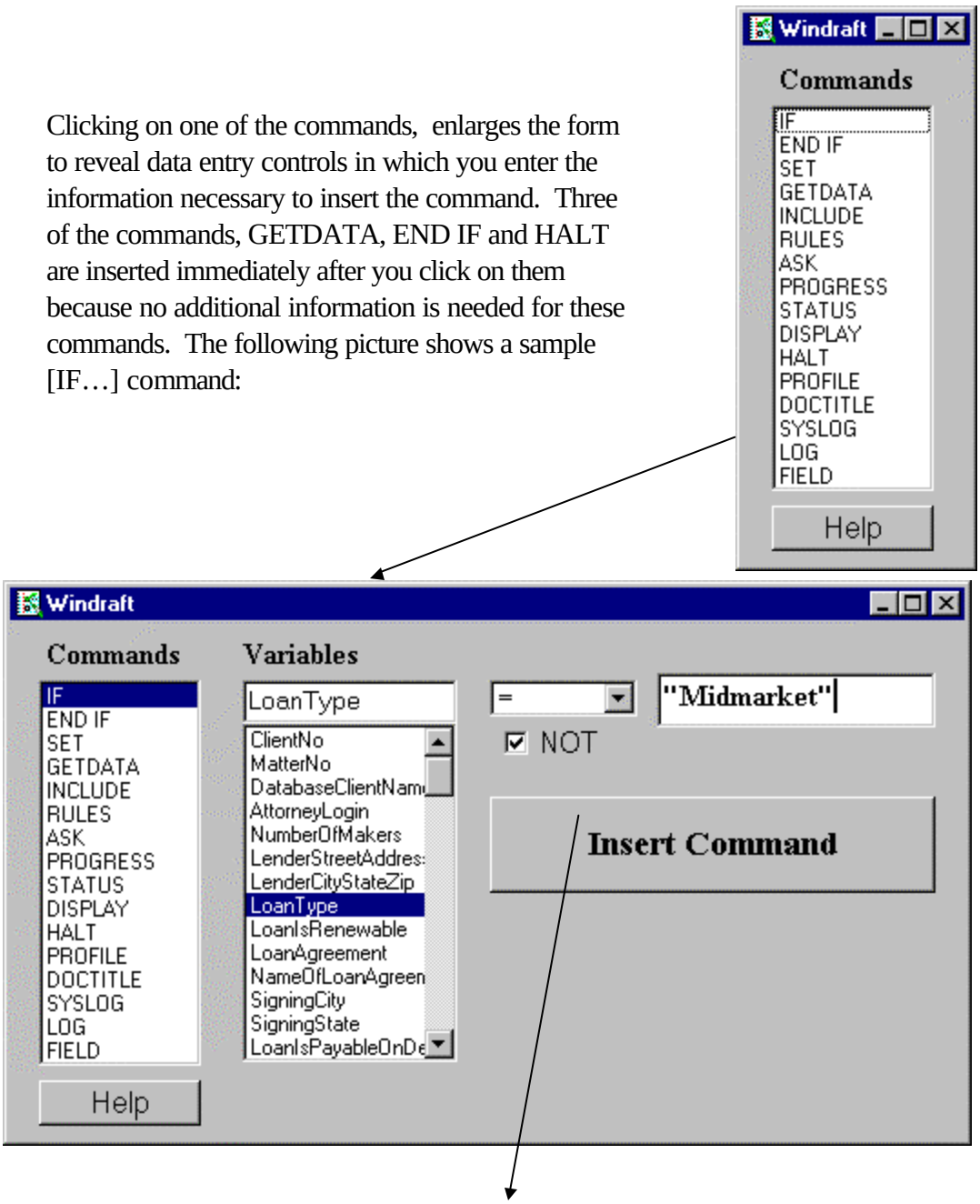


If you select Edit List…, VarList.txt is opened in Notepad so that you can alter the contents of the list file.

B. Insert Command

And to make using the Model Document Programming commands easier, there is now an automatic way to insert commands without having to write them out manually: the Insert Command dialog . This dialog also serves the purpose of clarifying command syntax.

Clicking on one of the commands, enlarges the form to reveal data entry controls in which you enter the information necessary to insert the command. Three of the commands, GETDATA, END IF and HALT are inserted immediately after you click on them because no additional information is needed for these commands. The following picture shows a sample [IF…] command:



This will insert **[IF NOT LoanType = "Midmarket"]**

XVI. Understanding Formatting in MS Word and WinDraft

A. How MS Word Defines Paragraphs

WHAT IS AN MS WORD PARAGRAPH?

As you enter text, MS Word will "wrap" at the ends of lines so that the text will continue on the next line rather than run beyond the right margin. Each such line is terminated with a "soft return" that brings the editing point back to the left margin. It is called "soft" because it will move around in the text if the margins are changed or words are inserted upstream. Pressing the <Enter> key, on the other hand, produces a "hard return" or "paragraph return" that also brings the editing point back to the left margin, but does not occur only at the right margin and will not disappear if the margins or the preceding text are changed. Two consecutive hard returns, or a soft return followed by a hard return, will produce a blank line. A paragraph, in MS Word, is all the text between two hard returns (or between the start of a word document and the first hard return). A paragraph in the usual sense is composed of sentences, is generally followed by a blank line and often has its first word indented. A paragraph in the MS Word sense need have none of these features. For example, the following lines:

War is hell.
War is
War
W

1234

are all (even the blank line) separate paragraphs from the point of view of MS Word.

B. MS Word Paragraph Attributes and Hard Returns

In MS Word, some attributes apply to characters or groups of characters (for example bold or font size), while others (such as centering, indenting and heading styles) apply to entire paragraphs. Word stores the latter paragraph attributes in the hard return at the end of the paragraph to which they apply. This raises an interesting question about what happens when a hard return is deleted. Two different results are possible depending how this deletion takes place.

1. Deleting a Single Hard Return

EXTENDING

When you combine two paragraphs by deleting the hard return(s) between

PARAGRAPH ATTRIBUTES

them, Word copies formatting information for the first paragraph to the next hard return down the line. The result is that the second paragraph will not only join the first but will assume the same paragraph attributes as well[1]. This is the effect that most people would expect. The formatting information for the first part of the new paragraph is extended and the attributes of what used to be the second paragraph are lost.

2. Deleting a Block that Includes a Hard Return

LOSING PARAGRAPH ATTRIBUTES

If you combine two paragraphs by deleting a block of text that happens to include (among other things) a hard return, MS Word will not look for or notice the deleted return and formatting information for the first paragraph will be lost. Consequently the resulting paragraph will assume the attributes of what used to be the second paragraph. This is *not* what the user generally expects.

C. How WinDraft Handles Paragraph Attributes When Merging

WinDraft's effects on paragraph formatting when merging model documents can be summed up in the following two rules which correspond, in a sense, to the two rules of the preceding section.

1. Returns After Closing Brackets May Migrate

FORMATTING IS EXTENDED WHEN PARAGRAPHS COMBINE

In merging, WinDraft deletes hard returns that come *immediately* after the closing brackets of commands that don't insert text (such as SET, IF, ENDIF, STATUS, etc.). If, in so doing, WinDraft combines two paragraphs of text then formatting information will be copied from the deleted return to the next one, as described in Rule 1 of the previous section. This will make the second paragraph take on the attributes of the first. If, on the other hand, a bracketed WinDraft command begins and ends on a line by itself then there is no text to combine, the hard return immediately after the bracket will not migrate and its formatting will be lost.

These different situations are illustrated in the following example. For clarity, spaces in the model document are shown as dots like·this, and hard returns are shown as ¶. This is how they appear in Word when you press the ¶ button or choose Tools|Options:View and check Spaces and Paragraph Returns under Non-printing Characters.

[1] This result does not apply to text in tables. In that context, deleting a hard return produces the result given in the second paragraph.

The first line of normal text in the example model document gets added to the heading in the merged document because the hard return at the end of the heading migrates. Similarly, the first bulleted item gets added to the second line of normal text when the return on the line after "apples" migrates. Notice that the entire paragraph gets indented because that was an attribute associated with the migrating return. The return on the line after "pears" does not migrate, because that SET command begins and ends on a line by itself. Finally, the return on the next-to-last line does not migrate because it is separated by a space from the closing bracket of the comment.

Model Document	***Merged Document***
A·Centered·Heading·[;·this·return·will·migrate-->]¶ This·is·normal·text.¶ Here·is·more·normal·text.[¶ SET·fruit·=·"apples"¶]¶ ?? This·is·an·item·on·a·bulleted·list.¶ [SET·fruit·=·"pears"¶]¶ More·normal·text.[;·this·return·will·***not***¶ migrate·because·of·the·space·preceding·it.✍]·¶ Another·bulleted·item.¶	A Centered Heading This is normal text. Here is more normal text.This is an item on a bulleted list. More normal text. ?? Another bulleted item.

2. Returns Within Command Brackets Are Lost Forever

FORMATTING IS LOST WHEN AN OPEN BRACKET ENDS THE PARAGRAPH

WinDraft will not merge any text or formatting within square brackets into the output document. Consequently, if a formatted paragraph in the model document ends with an opening bracket, the hard return for that paragraph will fall within brackets and be lost (together with the formatting information) in the merged document. Thus, if a WinDraft command is spread over more than one line (more precisely, if it includes hard return(s) within its brackets), it will produce the behavior described in Rule 2 of the previous section. The paragraph that ended with the opening bracket will take on the attributes of the hard return that appears next after the closing bracket.

This is seen in the third line of the previous example. The hard return at the end of the line "Here is more normal text" is lost, and replaced by the next hard return after the closing bracket, which was associated with the indent attribute.

D. How WinDraft Handles Character Attributes When Merging

WHAT HAPPENS TO

WinDraft may also modify character attributes (e.g. underline, font size and type, italic, etc.) when merging documents. Specifically, if the model

UNDERLINES AND FONT INFO WHEN MERGING

documents are all ".DOC" files, and the line ResetToStyles=1 is included in the WinDraft.INI file section for the current practice system, then all character formatting from the model documents will be lost. Instead, the merged document will use the styles defined in the ".DOT" file identified in the DefTemplate= line of that section of the WinDraft.INI file. If you wish to adapt existing, formatted documents for use with WinDraft, you probably want to set ResetToStyles=0. For more details, see Appendix IV: WinDraft.INI settings in the file WDAPPNDX.DOC.

If you are creating a practice system from scratch, it is recommended that you set ResetToStyles=0 and make the master model document a ".DOT" (Word template) file. This will ensure that character formatting in the master document will be retained, and will override that in subdocuments. In this way, complex practice systems with many subdocuments and written by several authors will still produce output documents with a consistent and defined style throughout.

XVIII. Tips for Better Model Documents

USE INDENTING TO ENHANCE VISUAL FLOW

To make your model documents more readable and easier to maintain, it is important to add comments liberally and to format your commands properly. All commands should be aligned with respect to the left-hand margin, indented appropriately. As mentioned in section 0: Basic Programming Concepts (page 13), **END** statements like **ENDIF** should be indented to the same extent as their matching **IF** command, and as command statements are nested within others they should be progressively indented. This is demonstrated in the programming example in the next section. Formatting commands for good programming style and formatting output documents correctly may, at first, appear to be conflicting imperatives. Both ends can be easily achieved, however, by keeping in mind the following simple rules.

A. How WinDraft Handles Whitespace

WHEN DO WinDraft COMMANDS PRODUCE BLANK LINES?

1. All whitespace (tabs, indents, spaces and paragraph returns) within command brackets will be ignored and will not affect the final formatting of the document

2. Paragraph returns *immediately* following a closing command bracket will not appear in the final document, either, except for commands that themselves generate text (INCLUDE, REF(), MERGEFIELD()) or variables (e.g. [LenderName]). This "vanishing command return" feature

means that a bracketed command on a line by itself will not produce a blank line in the output. If there is any whitespace between the closing bracket and the paragraph return, however, the return is no longer associated with the command. In that case it will not vanish, and will produce a blank line in the output. (Both were seen in the example on page 39.)

3. It is a consequence of preceding two rules that sometimes a paragraph of text in the model document will lose its paragraph return and the formatting associated with it.

4. When text is inserted into the merged document by use of the **INCLUDE** statement, a paragraph return is added after the inserted text. This is because every MS Word document ends with a paragraph return. Any hard return after the [INCLUDE "*filename*"] command will be in addition to this inserted return and would generate a blank line after the included text.

B. Proper Placement of Brackets

PLACE WHITESPACE BETWEEN BRACKETS TO FORMAT PROGRAM LOGIC

To solve the problem of unwanted formatting, place Word paragraph returns , tabs, and indents between the brackets in a programming command line. Often you will want to put returns and indents just inside the brackets in order to structure the commands so that it is easy to see the structure of the logic. This makes the logic much easier to understand. For example:

?? This is some text. I want more text to appear on this line without a hard return if the next IF statement is true. ***[***

IF YouWantText = "Y"
]This is the first optional text. If there were more optional text that you want right here, we would add it like this. ***[***

IF YouWant2ndOpt = "Y"
]This is the second nested optional sentence. ***[***
ENDIF][
ENDIF
]This is the rest of the paragraph after the optional part.

In the above example, because there are no tabs, indents, or returns outside the brackets (they are all contained within), the merged paragraph will be free

of any of this undesired formatting.

We strongly recommend that you take advantage of this formatting feature to make sure that **IF ... ENDIF**, and similar commands always visually line up so that you can see the matching commands and nested levels. It's also helpful to match corresponding **IF** and **ENDIF** statements by formatting them in the same color programming style.

C. Combo Codes

1. In General

Based on many years of programming practice systems, we have adopted this rule:

"The fewer variables, the better."

We found that it is much more difficult to program and maintain a document and checklist when there are lots of "Yes/No" fields rather than one field representing a number of pieces of information.

For this reason, we have adopted the practice of combining multiple sub-codes into one code. (Programming jargon for this is to "concatenate" the subcodes.) In an example drawn from our estate planning documents, special language needs to be included depending on whether the grantor is a trustee, whether the spouse is a trustee, whether a corporation is a trustee, *etc.* Rather than set up "Yes/No" variables for each type of trustee, we have set up just one *TrusteeType* code, which may have values like:

G	Grantor
GO	Grantor and Other
S	Spouse
GS	Grantor and Spouse
GSO	Grantor, Spouse, and Other

These codes can be combined in any order.

2. Using Combo Codes in a Document

"Operators" are used to evaluate the code to insert text into a document. For instance,

"=" , the "equal" operator

?? *[**IF** TrusteeType = "GS"]*
We can tell that there are two initial trustees: the Grantor and the Grantor's Spouse.
*[**ENDIF**]*

"C", the "contains" operator
?? *[**IF** TrusteeType **C** "S"]*
The Grantor's Spouse is one of the Trustees.
*[**ENDIF**]*

D. Math

WinDraft can use the power of Object Linking and Embedding (OLE) and the integrated functions of Microsoft Office to perform math calculations. WinDraft can tell Word to link to an Excel spreadsheet and imbed the whole spreadsheet, a group of cells or a single cell into your final document. To create this link, with Excel and Word open:

(1) In Excel, select the cell or cells and choose Edit|Cut;
(2) Return to Word; in your Word document, choose Edit|Paste Special Link; then, select Unformatted Text. As you make changes in Excel, your Word document will be automatically updated.

The Excel values can also be transformed and held as WinDraft variables by using the **SET** and **REF** commands.

Simple math calculations can also be done using WinDraft functions, or Word's tables. See above for documentation on WinDraft's functions, or refer to your Word manual to learn about Word tables.

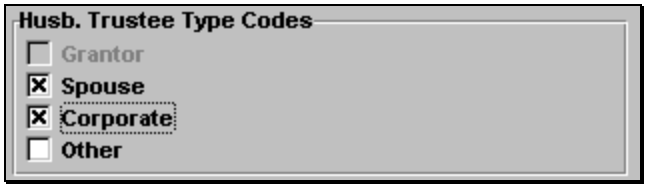
XIX. Tips for Better Checklists

A. Handling Combo Codes in the Client Checklist.

Combo Codes

WinDraft's WDData checklist language contains a special command for handling "combo codes." The **CODE** command is fully explained in the Checklist Template Programming Guide, but here is an introduction.

To the lawyer answering questions, the screen looks like a list of Yes/No check boxes, like this:



In the interface template, the system author lists the component parts of the code in a list, like this:

```
CODE TrusteeTypeCode "Husb. Trustee Type Codes"
      "G" "Grantor"
      "S" "Spouse"
      "C" "Corporate"
      "O" "Other"
END CODE
```

When the data file is saved, it combines the sub-codes into one text field, as described in the previous section. (For example, TrusteeType="GC", for Grantor and Corporate.)

XX. Troubleshooting Model Documents and Checklist Templates

WinDraft is very good at telling you where the specific error in your Model Document is located, thus enabling you to quickly correct the mistake.

A. Template Changes Are Not Appearing in the Checklist

If the author makes additions, deletions, or changes to the checklist template file and these changes are not reflected in the outline checklist, the problem is that these changes were not properly saved. Each checklist template is kept in two versions: one saved as a Word document for convenient editing and another, text only version with an ".WDT" extension. WDData reads only the latter. Return to the checklist template file (the version with the ".DOC" extension) and click on the above icon located on the toolbar. To perform this operation manually, do a File Save As and save the checklist template, not as a ".DOC" but as a Text Only file, giving it an ".WDT" extension.

B. Runtime errors while merging

1. Spacing

WinDraft is now very flexible in its handling of spaces. As long as there is at least one space character (space, tab, or return) between WinDraft keywords and arguments, it will interpret the command. It is easiest to see missing spaces if you turn on the display of spaces by clicking on the ¶ button on the tool bar.

2. Undefined Variable Referenced.

If you try to access a variable that is undefined, you will get this message. You will see it in a dialogue box during the merge and will also see "UNDECLARED: *VariableName*" in the document where the variable should have been. (This is not a disaster for fill-in-the-blank variables, but if it is used in an "**IF** statement," this error may cause the incorrect text to be inserted into the document.). WinDraft will also put a line about the error in the system log file.

By undefined, the program means that the variable in the Model Document either does not exist or differs from the variable in the checklist template. You may have a typographical error, so that the variable is defined with one spelling and referred to in the document with a different spelling. Note, however, that differences in capitalization do not matter.

To fix these errors, the variable name needs to be edited in the checklist template or the model document so that the two match.

If the undefined variable error message has attached a "$1" to the end of the variable name, then an argument in the Model Document which needs to be surrounded by quotes (such as a filename in an **INCLUDE** command or a variable that is part of a checklist **CHOICE** selection) is missing a beginning quote. Insert the appropriate quotation marks around the argument and the error is fixed. You must also surround "literal" text in an expression with quotation marks. For example, [**SET** *LastName* = "Jones"].

3. No End of Statement Bracket

One of the variables in the Model document is missing an end bracket. The next error message will show you which variable is missing the bracket. This error may also result in an Undefined Variable error. Write down the name of the troubled variable, go back and insert the

missing end bracket in the Model Document, and all of these errors will be fixed.

This error could also occur if two opening brackets were mistakenly typed. Again, WinDraft tells you the offending variable so that it can be easily fixed.

If the opening bracket of a variable is missing, no error message will be reported. Instead, the variable name and the end bracket will appear in the merged document. If you think this may be occurring, do an Edit|Find and search for a "]" in your merged document. If none is found, no problem exists.

4. Unmatched IFs and ENDIFs

If the Model Document has an extra **ENDIF** or **IF** statement, WinDraft will prompt the user with the suitable error message during the merging of the document. An **IF** without an **ENDIF**, for example**,** will give a "Missing ENDIF" error message. To correct these errors add the missing **ENDIF** or **IF** to the text as appropriate. If the Model Document is long, it is easier to spot these problems by printing out the Model document and matching the statements up. A failure to put a bracket before an **ENDIF** statement can also give this error as well as having a ";" on the line with the **IF** or **ENDIF**. As noted earlier, a "[;" tells WinDraft that this is a comment line, so ignore it. Thus, if your **ENDIF** or **IF** is on this line, it will be ignored and could result in an error.

5. Ending Quote Expected in Checklist Template

Unlike the above errors, this error occurs in the checklist template file, not in the Model Document. As the error suggests, one of the terms in the checklist template is missing a quotation mark. The error message reveals exactly which sentence is missing the end quote so that it can be easily found and fixed. Do not forget to save the new checklist template file as a Text Only file with an ".wdt" extension. Use this toolbar button to accomplish this task.

C. Unanswered

If the text of your merged document contains the word "UNANSWERED," then one of the questions that should have been answered at the data entry level

was left blank. Note, however, that this does not cause an error and the document will still merge.

To avoid this problem, always use the F4 keyboard command when filling out the data entry screens. This command takes you to all the relevant unanswered questions so that they can be answered. If all the questions have been answered, WinDraft tells you that no unanswered questions remain.

D. Formatting Problems

If you have a stubborn formatting problem, e-mail your file to Eidelman Associates and give us a call. We will be happy to help you solve it. Here are the most common ones:

1. Extra Paragraph Returns

WinDraft leaves in or deletes hard returns as it merges a document according to the rules set forth in the section called "A. How WinDraft Handles Whitespace" on page 40, above.

This means that if there is an extra hard return you don't want, it is probably one that looks as if it is on a line by itself in the Model Document, but isn't. (Perhaps it has a space between "]" and the return.) For this reason it is best to edit Model Documents with your options set to show spaces, tabs and paragraph returns. Do this in Word by pressing the icon, or by selecting Tools|Options; clicking on the View tab, then selecting the appropriate characters in the "non-printing" box in the lower right-hand corner of the dialog box.

2. Indents and Other Paragraph Formatting Problems

If you end up with paragraphs in the file that have paragraph formatting (heading styles, indentation, centering, etc.) but don't look right, it is probably because:

1. Word stores paragraph formatting in the hard return at the end of the paragraph, and
2. Based on the way WinDraft handles hard returns (see the section entitled "C. How WinDraft Handles Paragraph Attributes When Merging" on page 38, above), WinDraft has left in or deleted a hard return differently from the way you wanted. Go back and look at the Model Document. You may need to move a hard return inside or outside of a bracket, as the case may be.

3. "Body Text" Ends Up in a Heading

Look in the Model Document for the heading that got included after the problem text. One or more hard returns immediately before it are probably formatted in the same heading style. Reformat them in the correct style or delete them altogether and things should format correctly.

WinDraft

Checklist
Author's Guide

Eidelman Associates

317 South Division
Suite 187
Ann Arbor, MI 48104

734-769-1500

734-769-1501 (Fax)

73067.2131@compuserve.com

support@lawtech.com

Microsoft® SOLUTION PROVIDER

WinDraft Checklist Author's Guide

I. Checklist Author's Guide: Who Should Use this Manual

This manual is for authors who create data entry screens. It should be read **after** the WinDraft *Model Document Author's Guide*.

Reading this manual is not necessary to learn how to enter data and merge documents. Whether your documents have been set up by others in your firm or by Eidelman Associates, the WinDraft *User's Guide* will provide the instructions needed for routine document assembly.

This documentation is for the lawyer, paralegal, consultant or any other author who wants to create or edit the WinDraft Checklists (data entry screens and printouts). These are the screens that ask the user for client information (personal and legal). The information entered in data entry screens is merged with the Model Documents to create the customized client documents. The process of merging documents is described in the *User's Guide* (*see* **Blast Off**), and the process of programming Model Documents is described in the *Model Document Author's Guide*.

The best way to learn WinDraft's programming commands is to first become familiar with the process of entering data and merging documents. Then review both the Model Documents and the Checklist Templates provided with WinDraft as demo systems. Both of these (the Loan Agreement and Babysitting systems) possess demonstration forms and screens that use most of the programming commands you will need.

This manual assumes that you know how to do basic editing with Microsoft Word.

II. Purpose of Checklist Guide

This section provides instructions for editing existing or creating new checklists that form the WinDraft data entry screen. The term "checklist template" is used to refer to the "programming" used to create the WinDraft Checklist, (the visual outline that asks the user the appropriate legal and client questions.)

This manual uses the following layout:

?? Examples are preceded by bullets and follow the discussion of a command
?? *Variables* Items in italics are variable names.
?? **Commands** Items in bold are the example checklist template commands.

Checklist programming differs from Model Document programming in the following significant ways: (1) unlike Model Documents programming, Checklist does not require that variables and commands be in brackets; also, (2) Checklist programming requires that each command be put on its own line, whereas Model Documents allows you to put a command on multiple lines, as long as the command begins and ends with a bracket.

III. Concepts and Terminology

Checklists

A "checklist" is the WinDraft user interface for the data entry screen.

?? On the computer screen, WinDraft displays a "checklist" as an outline on the left side of the screen and a dynamic data entry screen on the right side. It uses standard Windows interface elements (text boxes, radio buttons, and check boxes).

?? On paper, WinDraft prints the outline, with headings indented and underlined, and the data entry screens appear just as they do on screen. This allows even computer phobic lawyers to use the system by working on paper.

Checklist Templates

A "checklist template" is a special kind of document that contains the programming commands to develop a checklist. When WinDraft loads, it reads the checklist template file, which is an ASCII text file on the disk. Authors edit and work with checklist templates right inside Microsoft Word. However, the checklist template is saved as a DOS text file, using a special button on the WinDraft Author Toolbar in Word .

Checklist templates can be quite simple if you don't want to program them with logic, or can be quite complex if you want to program them with legal rules that are interrelated or complex.

Programming a checklist template is very similar to programming a Model Document, but some different terms are necessary to create the questions WinDraft asks. There are less than 15 WinDraft checklist template commands. The main ones are all contained on a one-page insert, found in the front of this binder, that shows you the various commands needed to create the data entry screen questions . (See also, B. AUTHORING CHECKLIST TEMPLATE OUTLINES, pp. 43-44)

Your checklist templates are stored under your practice system in the Template directory. For example, the babysitting checklist template is stored as "*\WinDraft\babysit\template\babysit.wdt*".

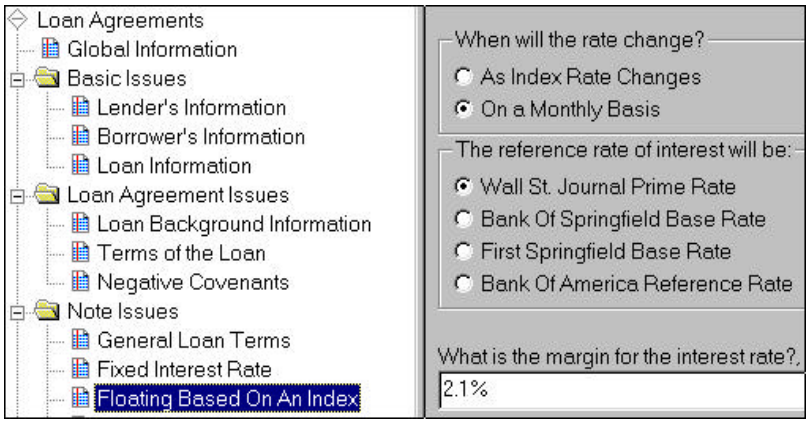
NOTE: Before the 2000 version of WinDraft (v4.0), the checklist templates were referred to as data entry 'scripts', and used a .int suffix. This usage

became confusing when ActiveScript support was added to WDData, so in later versions we renamed these to WinDraft Checklist Templates, which use a .wdt extension. (These are not to be confused with WinDraft Active Script files, which use a .wds extension). If you see any references to scripting that don't seem to be referring to VBScript or JavaScript, it is probably a result of this name change.

A. The Visual Outline

THE VISUAL OUTLINE

To provide an overview of the legal document, enabling the user to see the "big picture," the left side of the data entry screen is organized as an outline.



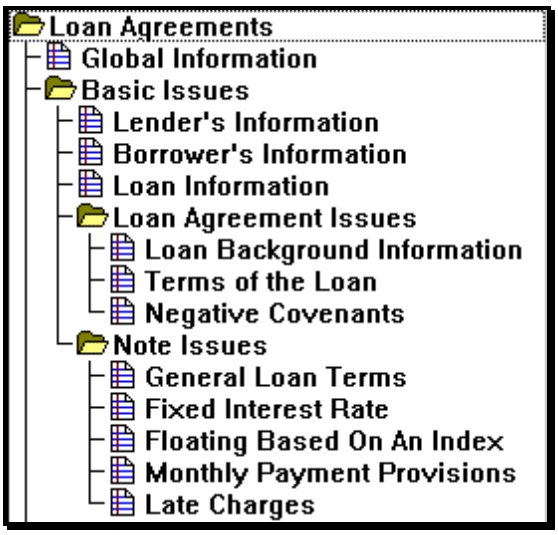
To create your own checklist template and outline, modify one of the demo checklist templates and/or use the Create New System macro (See *Using the New System Macro to create your own system from scratch.*) (The examples listed below show how to work with part of the screen shown above.)

B. Authoring Checklists: command summary & examples

Sample Checklist template with Outline Commands

```
OUTLINE Page1 "Loan Agreements"
HEADING1 Global "Global Information"
HEADING1 BasicIssues "Basic Issues"
  HEADING2 LenderInfo "Lender's Information"
  HEADING2 MakerInfo "Borrower's Information"
  HEADING2 LoanInfo "Loan Information"
  HEADING2 LoanIssues "Loan Agreement Issues"
    HEADING3 Background "Loan Background Info"
    HEADING3 TermsOfLoan "Terms of the Loan"
    HEADING3 NegCovenants "Negative Covenants"
  HEADING2 NoteIssues "Note Issues"
    HEADING3 GenLoanInfo "General Loan Terms"
    HEADING3 FixedInterestRate "Fixed Interest Rate"
    HEADING3 Floating "Floating Based On An Index"
    HEADING3 Payment "Monthly Payment Provisions"
    HEADING3 LateCharges "Late Charges"
END OUTLINE
```

Left side of data entry screen it creates



Commands with Examples		***Explanation***
SYSTEMNAME "Loan Agreements"	??	System Name command. This name will display when the user clicks on "Help|About WinDraft Data Entry"
COPYRIGHT "Jones & Smith 1995"	??	This will display a special Copyright notice for the practice system both on the system's title page on the right side of the screen and when the user clicks on "Help|About WinDraft Data Entry"
;; LoanAgmt.doc -- JAE last edited 12/19/95.	??	Comment (WinDraft ignores it) so that when people read the checklist template they have some audit trail information.
OUTLINE Page1 "Loan Agreement Data"	??	The on-screen outline MUST begin with the OUTLINE command, followed by Page1, and the text you want displayed next to the icon of a folder at the top of the outline.
HEADING1 Parties "Info about parties" **HEADING2** Lender "Lender Information" HEADING2 Borrower "Borrower Info" **HEADING3** PrimaryBorrower "Primary" HEADING3 Guarantors "Guarantors" etc. etc.	??	Each outline heading command line must have the following elements: 1. Heading level command (will be indented based on number) 2. Name of data entry screen PAGE that is displayed or printed with this outline heading. 3. Text to display or print in the outline heading, in quotation marks.
END OUTLINE	??	There must be an END OUTLINE command after all of the headings.
PAGE Parties TEXT variable "Prompt" YESNO variable "Prompt" CHOICE variable "Prompt" "option 1" "Text for Option 1" "option 2" "Text for Option 2" END CHOICE **END PAGE**	??	Each HEADING must have a matching PAGE and END PAGE, with the name of the page exactly matching the name in the heading. Between the PAGE and END PAGE commands are the data entry prompts. See *WinDraft Data Entry Screen Author's Quick Reference* for examples

PAGE Default
LABEL “Text you want on a default page”
END PAGE
PAGE Title
LABEL “Text you want on title page”
END PAGE

?? Each checklist template should have a page named “default” that will be displayed on the right side of the screen for any heading that doesn’t have a matching page.
?? When the checklist template is first loaded, the title page will display on the right side of the screen. It will display the name of the system, the copyright
?? and any other special information you want .

CREATING THE INDIVIDUAL PAGES

Each **HEADING** entry has a corresponding **PAGE** entry. A Page icon will contain the questions that are asked of the user during data entry.

Key Points of Page Construction:

It is not required, but we like to format the lines with **PAGE** commands on them with Word's outline heading styles. This makes them easier to read and to work with. See the Word user manual for more information on working with outline headings.

1. Each data entry screen must begin with a command like this:
PAGE *PageName*, where the page name matches the name in the heading command.

2. Each data entry screen must end with a command like this:
END PAGE

For instructions on how to "program" the data entry screens between the PAGE and END PAGE commands, see the examples and instructions that follow.

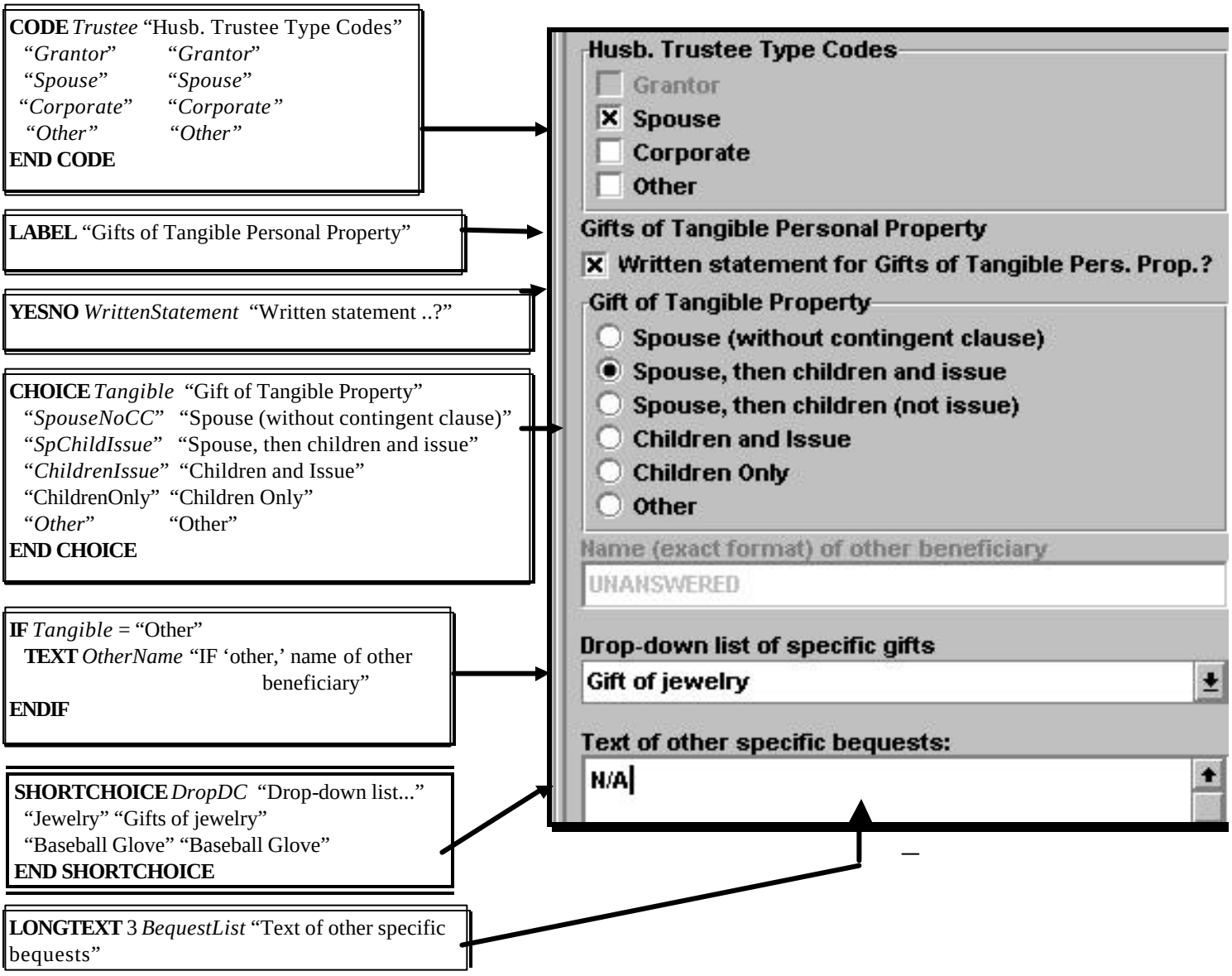
There are two "special" pages:

PAGE Title, which displays information on the title screen, and

PAGE Default, which displays information if a heading doesn't have a matching named page.

C. WinDraft Data Entry Screen

Author's Quick Reference -- Examples



Other commands that display: LINE – Displays a horizontal line.
SPACE – Extra blank line

TITLE – Same as LABEL, to display text, except that it centers the text.
EDITCHOICE - same as SHORTCHOICE except that you can enter text not on list.
MASKEDTEXT – same as TEXT, but only accepts data in a specified format
DATE – gives a drop-down calendar control on a TEXT box.

Other commands you can't see on the screen:

IF…ELSE…ELSEIF…ENDIF — Used to "gray out" questions that don't apply or to control the flow of logic. (See TEXT example above.)

SET — Used to set an answer, such as:
SET State = "New York"

D. Saving Your Checklist Templates

SAVING YOUR CHECKLIST TEMPLATE

Saving your checklist template files

Clicking on this icon on the WinDraft Author Toolbar saves your checklist programming, first as a word processing document, and then in a DOS text file format for WinDraft to read with a name that ends with ".INT". Finally, it closes the file so that WinDraft can run it.

Be sure to use this button to save and close your checklist template files. Do not save your files with the standard File|Save or Save As. Instead, click on this button and your work will be done for you. If checklist template changes you are making are not appearing in your data entry screen when you are running WinDraft, it is probable that you are forgetting to use this button to save your checklist template in text format.

IV. Basic Checklist template Command Language

This section describes the entire list of WinDraft checklist template or data entry screen programming commands that are used between **PAGE** and **END PAGE** to create one data entry screen. The important note is that WinDraft checklist template programming commands need not be enclosed in brackets. As a matter of clarity, we recommend you type all WinDraft checklist template commands in CAPITAL LETTERS. This makes the commands easy to pick out from variables and other text.

A. Asking Questions with WinDraft

WinDraft asks the user for client and legal information with four main commands—**TEXT**, **YESNO**, **CHOICE** and **CODE.** These, their variants, and the other WinDraft data entry commands will be discussed below. The variable name used in the commands must **exactly** match the variable name used in the Model Document. When WinDraft merges the document, it will pull the answer from the data entry screen and insert it wherever it sees this variable name in the Model Document. If the variables do not exactly match, WinDraft prompts the user with appropriate error messages, indicating exactly where and what the problem is. Fortunately, these errors can be easily fixed by matching up the variables in the checklist template and the Model Document.

As with the WinDraft Model Document authoring language, we recommend that variables be as long as necessary to completely describe them. Since they can not have spaces in them, it is good practice to capitalize the first letter of each word, like this:

```
ThisIsAVariableNameMadeUpOfANumberOfWords

Director1StreetAddress
```

Variable names often change several times over the development of a practice system. This is an evolutionary process, rather than something which occurs suddenly and all at once, and usually the same data files keep being used for testing. This can make debugging difficult when old variables are still present in your data files, but not in the checklist. When you change variable names, do an Edit | Remove Unreferenced in WDData to remove these orphaned variables, so you get an error message on document assembly (instead of your users getting the error message).

B. Commands that Ask Questions and Get Data

1. TEXT

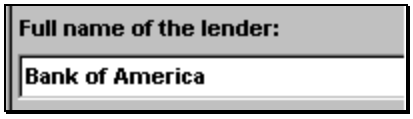
TEXT

The **TEXT** command is used to ask questions that require a fill-in-the-blank response.

?? **TEXT** *VariableName* "Prompt for user" *VariableName* represents the data variable that will be stored in the data file, and the text in quotes reflects the question that will be asked of the user.

For example:

?? **TEXT** *LenderName* "Full name of the lender:"
Typing this in creates the picture shown below. "Bank of America" was typed in by the user to answer the question.



NOTE: A text command can hold up to 8000 characters. A long answer will scroll out of the box, like a ticker-tape, but all of your information will be properly saved in the data file and inserted into the final document.

The TEXT command has several variants:

LONGTEXT

The **LONGTEXT** command is another command to use when you want a

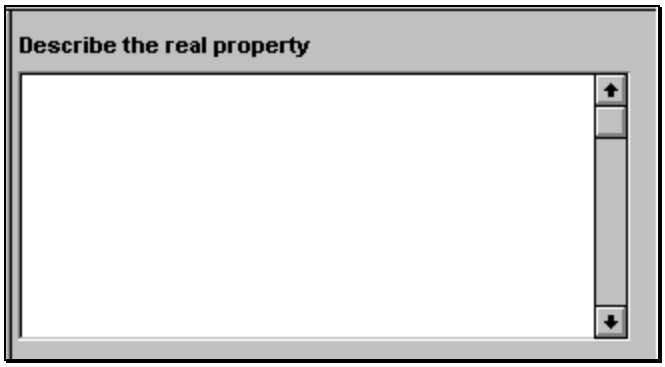
fill-in-the-blank response at the data entry screen. In functionality, it is the same as the **TEXT** command above, but it provides a different visual appearance. For example, a question asking for the description of property may require multiple lines of text.

?? **LONGTEXT** *NumberOfLinesToDisplay VariableName* "The question you want a **LONGTEXT** response to".

A **LONGTEXT** question can also accept up to 8000 characters. It can also accept returns, which will then appear in the output document.

Here, the number of lines defines how many text lines you want to leave available for the description. In the example below, eight lines were selected, and the picture below was created:

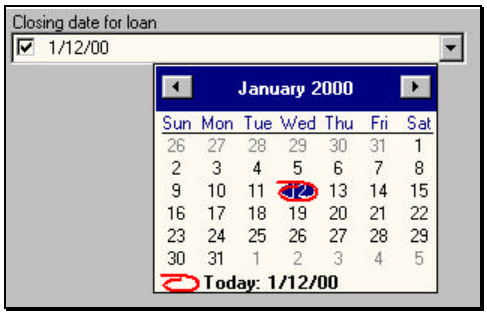
?? **LONGTEXT** 8 *DescriptionOfProperty* "Describe the real property"



DATE

The **DATE** command is similar to the **TEXT** command, except that a drop-down calendar is displayed to the right of the control. The syntax is identical.

?? **DATE** *ClosingDate* "Closing date for loan"



MASKEDTEXT

The **MASKEDTEXT** command works exactly like the **TEXT** command, except a "mask" is specified for what kind of data will be accepted. This is useful for ensuring the user enters phone number, currency, and other such structured data in the desired format.

?? **MASKEDTEXT** *VariableName* "Mask" "Prompt for user"

For example:

?? **MASKEDTEXT** *PhoneNumber* "(#{3})-#{3}-#{4} #@" "Enter a phone number"

Will create the following box:

and make sure that the user enters the phone number only in the format (###)-###-#### (with an optional extension afterwards).

The pattern for the mask is very similar to that used for Pattern Matching in Edit | Find in Microsoft Word.

Mask definition:

\	'Escapes' the following character, that is, treats it as a normal character and not as the beginning of a mask command.
?	Forces entry of exactly one character (any character)
*	Allows entry of any string of characters (of any length, containing any character)
#	Allows entry of any digit (0-9)
[ ]	Allow entry of whatever characters are specified between the

brackets. Ranges may also be specified (in ascending ASCII order), using a hyphen. For example, [0-9,;] allows entry of any digit, a comma, or a semicolon.

Alternately, if the first character inside the brackets is an exclamation point, it will allow any character NOT specified in the brackets. For example, [!A-Za-z] will allow entry of any non-alphabetic character.

@ Modifies the previous expression to allow zero or more entries of it. For example, #@ allows entry of a string of digits of any length.

{n} Modifies the previous expression to allow exactly n entries of it. For example #{3} requires 3 digits.

{n,} Modifies the previous expression to allow at least n entries of it. For example #{1,} requires at least 1 digit, but accepts as many digits as the user might enter.

{n,m} Modifies the previous expression to allow at least n entries, but at most m.

Other characters are treated as literals, and will come straight through into the textbox.

Examples:

(#{3}) #{3}-#{4}	phone number, e.g. (734) 769-1500
#{1,}.#{2}	currency entry (without commas), e.g. 1000.00
[!\@]\@[!\@]	email address, e.g. someone@somewhere.com
#{2}/#{2}/#{4}	date, e.g. 07/03/1975

Differences from the Word patterns:

Parentheses are not supported. The @ symbol allows 0 or more entries, rather than one or more. The # character allows entry of any digit.

Variable length fields may have an ambiguous ending point. However, the *first* character which satisfies the expression following the variable length expression is treated as the end of this field.

For example, in the email address mask given above, let ^ represent the current cursor position. If ^rsw@lawtech.com is in the MASKEDTEXT element, and the user goes to the beginning and starts typing, it will shift things right. Eg, eidelman^rsw@lawtech.com. As soon as the user presses the @ key, everything following the cursor position up to the current @ position is deleted, and entry continues, eg eidelman@^lawtech.com.

2. YESNO

YESNO

The **YESNO** command displays a check box and is used when a question calls for a Yes or No response.

?? **YESNO** *VariableName* "Prompt for user"

For example:

?? **YESNO** *SecurityDeposit* "Will a security deposit be required?"--

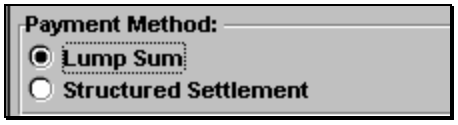
The above creates the box shown below:

Remember, a check corresponds to a "Yes" response

3. CHOICE

CHOICE

Use a **CHOICE** command when a checklist question calls for a selection to be made from one of several mutually exclusive options. The **CHOICE** command displays a prompt and "radio buttons" shown below.



The command to create this example looks like this:

CHOICE *PaymentMethod* "Payment Method"
"Lump Sum" "Lump Sum"
"Structured Settlement" "Structured Settlement"

END CHOICE

As you can see, there are three parts to the **CHOICE** command:

1. The first line contains the **CHOICE** command, followed by the name of the variable assigned to the question that is being answered and the prompt to the user for the question as a whole.

2. The middle section contains one line for each radio button you want to be displayed. Each line contains two text strings in quotes. The first is the

answer that will be assigned to the variable if that button is pressed, and the second is the prompt that is displayed next to the button. You can have as many buttons as you wish.

3. The last line is the **END CHOICE**, which is used to tell WinDraft to close this set of buttons. (The **END** is required. The word **CHOICE** is optional, but recommended for clarity.)

Here is a generalized example:

?? **CHOICE** *VariableName* "Prompt for question as a whole"
"*Answer1*" "Prompt for first button"
"*Answer2*" "Prompt for second button"
"*Answer3*" "Prompt for third button"
?? **END CHOICE**

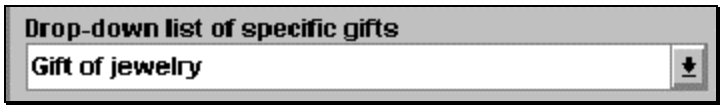
WinDraft doesn't care whether you use tabs or spaces, or how many of them, to format the answers. However, it is recommended that you indent the answers within the **CHOICE** ... **END CHOICE** to make them easier to read.

There are several variants of the CHOICE command:

SHORT-CHOICE

EDITCHOICE

The **SHORTCHOICE** and **EDITCHOICE** commands have the same syntax as the **CHOICE** command. Instead of displaying in a set of radio buttons, however, these commands give a drop-down list.



SHORTCHOICE allows the user to select only items which are on the list. **EDITCHOICE** allows the user to choose any item on the list, or enter her own.

The **SHORTCHOICE** and **EDITCHOICE** commands also support presenting lists formed from a database query. Define a query in the **DEFINITIONS** section (described elsewhere), and reference it in the **SHORT/EDITCHOICE** command. For example:

SHORTCHOICE Husband.Key
QUERY People
ENDCHOICE

Restrictions:
Since Windows doesn't support graying out items in a drop-down list, **IF** logic is not supported inside a **SHORT/EDITCHOICE** command. (You can gray out the whole command, but not individual choices).

EDITCHOICE requires that the prompts and the values assigned be the same.

4. CODE

CODE

Use the **CODE** command when your answer could have more than one selection that is true.

For example:

The syntax used to create this question looks like this:

?? **CODE** *Topping* "Topping for your sundae:"
"J" "Jimmies"
"*WC*" "Whipped Cream"
"*HF*" "Hot Fudge"
"*Oreos*" "Oreos"
?? **END CODE**

The formatting is exactly like the **CHOICE** command, described above. The difference is that **CODE** allows the user to select none, one, or as many of the options as he or she wants.

Note that what WinDraft does with the answers is to "concatenate" them. This is a computer term of art that means "to string together." For example, if you concatenate "Hi, " with "there!", the result is "Hi, there!" Thus, in the example above, the field Topping in the data field (and in the document) would be "JWCOreos", which is the concatenation of the answers selected.

This is most often used in the WinDraft Model Documents with the operator "C", which means "contains." In this case, in the Model Documents, if you want to know if the user selected "HF" for hot fudge, there could be an IF

statement like this:

?? **IF** Topping C *"HF"*
etc. ...

5. BUTTONs

BUTTON

The **BUTTON** command is used by advanced WinDraft programmers to active VB/Javascript programs.

?? **BUTTON** *ButtonName* "Prompt for user"

When any button is pressed, the OnButtonClick event is called in VB/Javascript, with *ButtonName* as an argument.

?? **BUTTON** *Insert* "Insert"
?? **BUTTON** *Delete* "Delete"

You may put up to three buttons in a row, and WinDraft will place them on a single line (as shown in the Insert/Delete example above).

C. Commands to Format or Explain

1. LABEL

LABEL

The **LABEL** command is used to display text on the screen without asking a question, allowing a response or affecting the data file. It is often used as a heading or to clarify meaning.

For example:

is created by this command:

?? **LABEL** "Lender's Information:"

Expressions (including variables) are supported in the prompts of LABELs and TITLEs.

?? **LABEL** "Information for " + GrantorName

2. TITLE

TITLE

The **TITLE** command is exactly the same as the **LABEL** command, except that the text is centered on the screen, instead of being left justified. This is typically used on title pages for title lines and copyright notices.

Overuse of **TITLE** elements may make your screens hard to read. For this reason, it is usually preferred to use **LABEL**, rather than **TITLE**.

3. ; Comments

";"

The "**;**" is used in the checklist template (without the quotes) to designate a comment, when the author wants to leave a note to himself or to others. WinDraft disregards everything on a semi-colon line, including what would normally be programming commands. The semi-colon only controls the line it is on, and if text exceeds the length of the line, a new semi-colon must be used on each line. Comments should be used liberally.

In a complex document, semi-colons should be generously used to keep everyone informed of what is going on in the programming and why. As many semi-colons as you want can be lined up in one line.

?? **;** This checklist template for corporate documents was written by CCM
;; It was revised 4/1/95
?? **;;;;;**This is to show you that you can use multiple semi-colons to make
;;;;;;it easier to spot them, and also to show that when your comments
;;;;;;; carry over to the next line, a new semi-colon must be used.

4. SPACE

SPACE command

The **SPACE** command is used when the author wants to insert a blank line on the data entry screen. This is purely a formatting command to improve the appearance of the data entry screen.

?? **SPACE**

5. LINE

LINE command

The **LINE** command is used when the author wants to insert a horizontal line on the data entry screen. This is another formatting command to improve the appearance of the data entry screen

?? **LINE**

D. LOGIC COMMANDS

The following commands control the "thinking" of WinDraft.

1. IF/ENDIF

IF/ENDIF command

The **IF/ENDIF** command is used the same way in the checklist template as it is used in the Model Document. Whenever an **IF** statement is used, a corresponding **ENDIF** must be placed at the close of the statement.

IF instructions test whether one or more conditions are true. When the condition set up in the statement is true, WinDraft acts on the command lines between the **IF** and the **ENDIF**

When the condition is false, or is irrelevant based on previous statements, WinDraft moves on to the next instruction and the skipped information is displayed on the data entry screen in gray.

?? **IF** *PaymentMethod* = *"LumpSum"*
    TEXT *SettlementAmount* "The lump sum settlement is:"
ENDIF

This example also shows how the **IF** statement was used in conjunction with the **CHOICE** option. The **IF** statement checked the answer to the above **CHOICE** question, and if the user chose the "Lump Sum" option, then this question will also be asked. If the structured settlement option was chosen instead, this question appears in gray on the Checklist.

?? **IF** *SecurityDeposit* = "Y"
    TEXT *AmtOfSecurityDeposit* "The security deposit amount is:"
ENDIF

The above example shows how the **IF** statement can be set to turn on based on the response to our **YESNO** commands (Y) or (N). Although all of these examples have just one question underneath, this was merely done for simplicity. As many questions as needed, as well as additional **IF/ENDIF** statements (each with their own **ENDIF**), can be located under an **IF** command.

Finally, as in the Model Document, **IF** statements can be nested as deeply as required to carry out the necessary commands.

ELSEIF and ELSE

ELSEIF and ELSE are used in checklist programming in the same way they are used in model document programming. For example:

```
?? IF SecurityDeposit = "Y"
      TEXT AmtOfSecurityDeposit "The security deposit amount is:"
   ELSE
      ; Questions to be asked if there is no security deposit
   ENDIF
```

IF logic in the outline to hide data entry screens

IF statements can also be used in the OUTLINE section of the checklist template. Instead of graying out everything on the pages inside the IF, WinDraft hides the associated checklist pages. Such pages will also be suppressed from the printout.

Please use cautiously, as it can be confusing to users when pages appear and disappear too often. Also, take care to put the question that hides a page on a page other than the one that is hidden. That is, don't put the "Are there guarantors?" question on the "Guaranty Information" page.

2. SET

SET command

The **SET** command is used similarly in both checklist template programming and Model Document programming. The **SET** command can be a powerful tool for assigning values to variables based on a user's preceding answers to questions.

For instance, with the **SET** command, you can assign a value, based on a user response, to a variable.

The **SET** command can be located anywhere in the checklist template.

There are some significant limitations, however:

- Only one **SET** command can be active per variable. You can have multiple **SET** commands in **IF** blocks, as long as only one will be used at one time.
- If you use **SET**, you cannot prompt the user for that variable. It will be answered, but will be grayed out. If the **SET** is in an **IF** block, and becomes inactive, however, the variable will revert to

it's previous value and become ungrayed.

3. NOT

NOT modifier

The **NOT** modifier can be used with an **IF** statement to test whether something is not true.

```
?? IF NOT LenderEntityType = "Individual"
      TEXT StateOfIncorporation "Where was the entity organized?"
   ENDIF
```

In the above example, if the user responded that the Lender was not an individual, then WinDraft would also ask for the state of incorporation of the business.

4. AND

AND modifier

The **AND** modifier is used when the insertion of a paragraph or provision hinges on several conditions being met.

For example,

```
IF BorrowerIsMarried = "Y" AND SpouseWillCosign = "Y"
   TEXT NameOfSpouse "Enter the name of Borrower's Spouse"
ENDIF
```

5. OR

"OR" Modifier

The **OR** function operates the same in the checklist template program as it does in the Model Document program. It allows a question to be inserted or asked if one of several conditions is met. An example of the use of **OR** is:

```
IF EntityType = "Corporation" OR EntityType=
"LimitedLiabilityCompany"
      TEXT StateOfIncorporation "Company is incorporated in:"
ENDIF
```

When using the **OR** command, the variable name that is being checked for must appear on both sides of the **OR**, as shown above.

6. ">", "<", "C", and "=" operators

">", "<", "C" and "=" operators

The usual operators may be used with **IF** statements that compare two expressions:

=	Equal To
>	Greater Than
<	Less Than
<>	Not equal to
C	Contains text string

The ">," "<" and "="commands can also be used when writing checklist templates. They would typically be used in conjunction with an **IF** statement to test whether a condition has been satisfied.

```
?? IF NumberOfChildren > "1"
      TEXT NameOfChild2 "The name of the second child is:"
        IF NumberOfChildren > "2"
          TEXT NameOfChild3 "The name of the third child is:"
        ENDIF
ENDIF
```

E. Special commands

There are a number of commands that don't fit into any of the categories above:

SYSTEMNAME

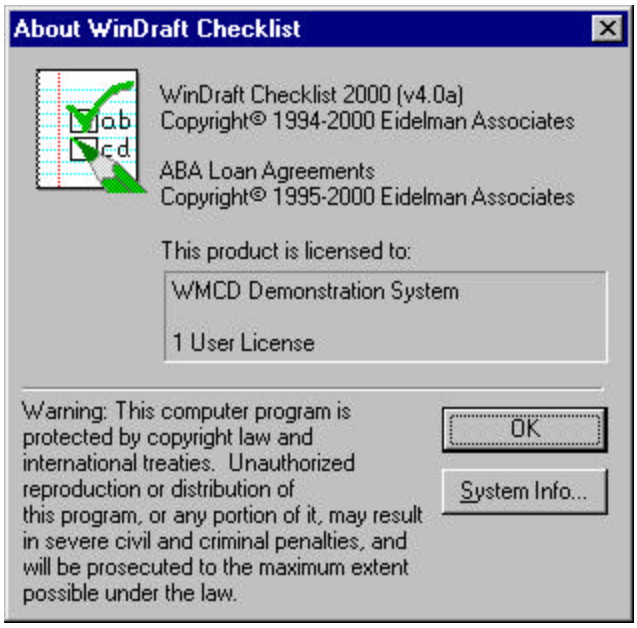
COPYRIGHT

The **SYSTEMNAME** and **COPYRIGHT** commands determines what text should appear in the about box. Use them to describe your custom practice system and provide a copyright message.

For example:

SYSTEMNAME "ABA Loan Agreements"
COPYRIGHT "Copyright© 1995-2000 Eidelman Associates"

would give the following messages in the about box:


About WinDraft Checklist
WinDraft Checklist 2000 (v4.0a)
Copyright© 1994-2000 Eidelman Associates
ABA Loan Agreements
Copyright© 1995-2000 Eidelman Associates
This product is licensed to:
WMCD Demonstration System
1 User License
Warning: This computer program is protected by copyright law and international treaties. Unauthorized reproduction or distribution of this program, or any portion of it, may result in severe civil and criminal penalties, and will be prosecuted to the maximum extent possible under the law.
OK
System Info...

ACTIVESCRIPT

The **ACTIVESCRIPT** command tells WinDraft Checklist to use active scripting to program the practice system. It takes an optional argument of what scripting language (supported by Microsoft Active Scripting) to use.

ACTIVESCRIPT
ACTIVESCRIPT VBScript
ACTIVESCRIPT JavaScript
ACTIVESCRIPT PerlScript

If no argument is given, then WinDraft defaults to using VBScript. The script file is loaded from the same path an filename as the checklist template, but with an extension of wds. For example, if your checklist template is C:\Program Files\WinDraft\Babysit\Babysit.wdt, then an **ACTIVESCRIPT** line will load a VBScript file from C:\Program Files\WinDraft\Babysit\Babysit.wds.

See the section on Active Scripting for more information.

DEFINITIONS section

The **DEFINITIONS** section, while not a command itself, groups a number of other commands together. It is similar to the **PAGE** and **OUTLINE** sections, but takes different command (described below).

DEFINITIONS
  VARIABLE VariableForScripting

DATASOURCE WDPeople “DSN=WDPeople”
QUERY People WDPeople “SELECT Key, FirstName & ‘ ‘ & MI & ‘ ‘ & LastName FROM Persons ORDER BY LastName”
END DEFINITIONS

VARIABLE

The **VARIABLE** command goes in the **DEFINITIONS** section. It defines a variable, and ensures it never gets marked disabled for blastoff. This is useful if you have VB or JavaScript code which manipulates one or more variables.

If you have variables which are not declared with the **VARIABLE** command, and which are not associated with an enabled (not grayed out) control on some page of your outline, then they will be given the value “DISABLED” on blastoff.

DATASOURCE

The **DATASOURCE** command goes in the **DEFINITIONS** section, and is used to define a database for later use in queries.

DATASOURCE Db2 "DBQ=db2.mdb;DRIVER={Microsoft Access Driver (*.mdb)}"
DATASOURCE WDPeople “DSN=WDPeople”

The command takes two arguments. The first is a name for the database (which is later used in **QUERY** commands). The second is an ODBC connect string, used to find the database. Please see the ODBC documentation for more information on connect strings.

QUERY

The **QUERY** command goes in the **DEFINITIONS** section, and is used to define a query on some **DATASOURCE**, which can then be referenced from a **SHORTCHOICE** command.

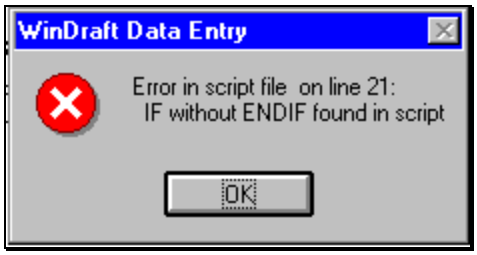
QUERY People WDPeople “SELECT Key, FirstName & ‘ ‘ & MI & ‘ ‘ & LastName FROM Persons ORDER BY LastName”

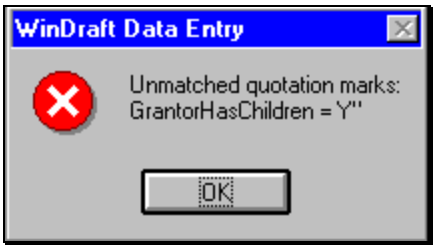
The **QUERY** command takes three arguments. The first is a name for the query, later used to reference it from a **SHORTCHOICE** or other command. The second is the name of a database, previously defined with a **DATASOURCE** command. The third is an SQL query, which tells WinDraft what information to draw from the database.

Please refer to the SQL documentation for more information on the SQL language, or use a tool like MS Query or Access to build your query.

V. Troubleshooting

If there are checklist template errors, WinDraft will generally tell you on what line number the problem is appearing.





To quickly get to errors, (1) get into Microsoft Word, (2) use File|Open to load the checklist template file on screen, (3) choose Edit Go To from the pull down menu, (4) select Line, and (5) type in the line number where the error is. WinDraft may also tell you exactly what the error is, as in pointing out that an ending quotation mark is missing.

The following are some potential errors and their solutions:

A. The changes you made to the checklist template don't appear in the checklist

For some reason, the changes you made weren't saved to the ".INT" file. Either you exited without saving, or you just saved the checklist template in Word but didn't also save it as a text file. You should save and close the checklist template file by using the special Save Checklist button on the author toolbar, which will automatically save the DOC file and the INT file and close both. It is on the far right, with a picture of a disk and two checkmarks.

B. "Page names must be less than 64 characters"

The internal name given to a checklist template page with the **PAGE** command must be less than 64 characters. Resolve the problem by shortening and renaming the page in the **HEADING** and at the corresponding **PAGE.**

C. "Unrecognized Command"

A command exists that WDData, the program which controls the data entry screen, does not know how to handle. It is possible that this is an appropriate WinDraft command, but it is being used out of context. For example, a **LABEL** command can

only be used inside a **PAGE** section of the checklist template. To avoid repetitious message boxes, WinDraft only displays the first error of this type when starting up.

D. "IF without ENDIF found in checklist template"

WinDraft made it to the end of a checklist template **PAGE** without finding an appropriate **ENDIF** to match an **IF** statement within that page. The line number that WinDraft provides as a reference mark for the error gives you the line number at the end of the page, rather than the actual line from which the **ENDIF** is missing. No actual line number can be given because WinDraft cannot be sure where you wanted the **IF** statement to end.

E. "Number expected in LONGTEXT"

The syntax of the **LONGTEXT** command is:
?? **LONGTEXT** *Number VariableName* "Question to be asked"
where *Number* is how many lines of blank text should be displayed at the data entry screen; *VariableName* is the WinDraft variable to associate the control with, and "Question to be asked" is the text to display above the control which asks the user for the response. *Number* must be numeric, and greater than zero. Note that *Number* should also be small enough so that it is easy to work with. For example, A **LONGTEXT** with 12 lines may be cumbersome for the users of your checklist template.

F. "Unrecognized PAGE command"

An inappropriate command was used within a **PAGE** section. Although the command may be valid in other parts of the checklist template, it is not supported within a **PAGE** section. For example, a **HEADING** command is valid in the **OUTLINE** section of a checklist template, but not in a **PAGE** section.

G. "Unexpected End of Checklist Template File"

The author failed to supply a matching **END** statement to a **PAGE** or **OUTLINE** section. All such sections must be ended with the appropriate (**END PAGE**, **END OUTLINE**) **END** command.

H. "Invalid heading level for outline"

All heading levels need to be between zero and nine. Re-define your outline so that it is not nested nine levels deep. You can use as many folders and pages as you need, they just cannot be nested this deep. Even in programming very sophisticated will and trust drafting systems, this limit was never reached. It is probably just a typographical error.

I. "Unrecognized outline command"

An inappropriate command was used within an **OUTLINE** section. Although the command may be valid in other parts of the checklist template, it is not supported within

the **OUTLINE** section. For example, a **LABEL** command is valid in a **PAGE** section, but not in the **OUTLINE** section.

J. "Checklist template line too long"

Checklist template lines used to program the data entry screen must be less than 1000 characters. They should also be *much* shorter than this, as a matter of good programming. Extremely long lines are confusing and hard to maintain. Again, even when programming complex systems, this limit has never been reached.

K. "Ending quote expected"

An ending quotation mark was left out in the checklist template. For example, the following line would give this error:

?? **TEXT** *ErrorVariable* "This has no quote at the end

The above would cause WinDraft to prompt with the suitable error message that is fixed by typing in the quotation mark at the appropriate location.

L. "Second quoted argument expected"

A command which requires two strings enclosed in quotation marks didn't find the second one.

VI. Other Checklist Template Errors

A. "No such page exists"

A page referenced in the **OUTLINE** section does not exist in the checklist template. To prevent this error, you can create a default page which will be used whenever a page cannot be found. To correct this problem, check for typos between the name used in the **OUTLINE** section and **HEADING** and the appropriate names of the corresponding **PAGES**. Also, make sure to name all your **PAGE's** and **HEADING's**. If you are getting this error and are sure that all of the pages in your outline are in your checklist template, make sure that you have a title page.

B. "Unmatched quotation marks"

This error is given for **IF** and **SET** statements that are missing a necessary quotation mark. Again, WinDraft displays the line with the problem so that the error can be easily spotted and retrieved by performing a Edit Find, in the checklist template, with the word processor. For example, this error would occur if the following line were in the checklist template:

?? **IF** *GrantorHasChildren* = Y"

Note how the opening quotation mark is missing.

C. "Two conflicting SET's were issued"

Two **SET's** were active at the same time and acting on the same variable. WinDraft doesn't know which one the user wants to be controlling, so it disallows the second **SET** command, and prompts the user with an error message so that the correct command can be chosen.

D. "Non-existing page reference"

This error is probably a symptom of extremely low memory. Restart Windows and try again. If this error persists, call Eidelman Associates at (313) 769-1500.

E. "There are no outline nodes specified in the checklist template. WDData requires at least one node on it's outline."

Your checklist template lacks an **OUTLINE** section or has an invalid **OUTLINE** section. Adding the section will alleviate this problem.

VII. Other Errors and Error Messages

A. "The checklist template you have loaded is for the practice system*..."*

You have changed practice systems and are attempting to blast off with data in a checklist template from the old practice system. WDData warns you of this, although it allows you to proceed if is what you really want to do. Note that attempting to proceed will probably generate errors in the output documents.

B. "No enabled controls are unanswered"

This message is given when a user tries to do an Outlined Find Next Unanswered (F4) and all of the questions in WDData are either answered or disabled. This should be looked at more as a message that the data entry screen checklist has been completed than as an error message.

C. "Unable to save to DOCS OPEN"

This error should occur only if you are using the DOCS OPEN document management software. This indicates that WDData was unable to save a file to the paths which DOCS OPEN specified. This is probably an error in DOCS OPEN. Try restarting DOCS OPEN and/or Windows to fix this.

D. "Unable to save blast off file"

WDData was unable to save a temporary data file for 'blasting off' into Word. The most common reasons why this occurs are (1) a lack of disk space or (2) an invalid directory specification in the ".INI" file.

E. "Practice System is not set, or set to a system without a WDData checklist template."

Either you have not selected a practice system, or there is no entry in the ".INI" file for the checklist template for this system.

F. "Unable to open WDData checklist template"

WDData was unable to open the checklist template specified in the ".INI" file for the current practice system. This may indicate that the file doesn't exist, or that the path is bad.

G. "Unable to create the outline control"

This error should occur only under extremely low memory conditions. Restart windows and try again. If this error persists, contact Eidelman Associates.

H. "Unable to open file"

The file specified in the most recently used file list no longer exists.

I. "The DOCS OPEN support in WDData is disabled. Cannot load the DOCS OPEN data file."

An attempt was made to open a WDData file from the DOCS OPEN desktop via a DDE command, but DOCS OPEN support is turned off in WDData's ".INI" file. If you wish to use DOCS OPEN, enable it in WinDraft.INI.

J. "Unable to load the Microsoft System Information program"

The MSINFO Microsoft System Information executable is missing, or not properly setup.

K. "Out of resources"

This error should only occur under extremely low memory conditions. Please restart Windows and try again. If the error persists, call Eidelman Associates.

L. "This EXE file has been tampered with. Please get a fresh copy of WDData."

The WDData exe file has been changed. Please reinstall from your original disks.

M. "Unable to access DOCS OPEN API. Please load DOCS OPEN."

WDData is unable to access the DOCS OPEN API. This probably indicates that DOCS OPEN is not loaded. Please load DOCS OPEN and try again.

N. "Unable to display control*"*

This error should occur only under extremely low memory conditions. Please restart Windows and try again. If the error persists, call Eidelman Associates.

VIII. Conclusion

WinDraft is a very powerful document drafting engine. After a few hours, all of its terms can be understood and the process of programming Model Documents and checklist templates can be started. Begin with simple documents; then, progress to the complex.

The underlying concept behind WinDraft is that it takes legal ability to successfully utilize the program, not computer skills.

To facilitate the learning of WinDraft and to see concrete examples of a Model Document, samples have been included. Although these examples are simple, they contain nearly all of the WinDraft "programming terms." When designing new systems, consult them and follow their format. If you do, you should not encounter any problem that cannot be quickly fixed after consulting the troubleshooting part of the Model Document Programming Guide.

I. Overview of Automating Your Forms

A. Background Required

Before you attempt to create your own systems, you should:

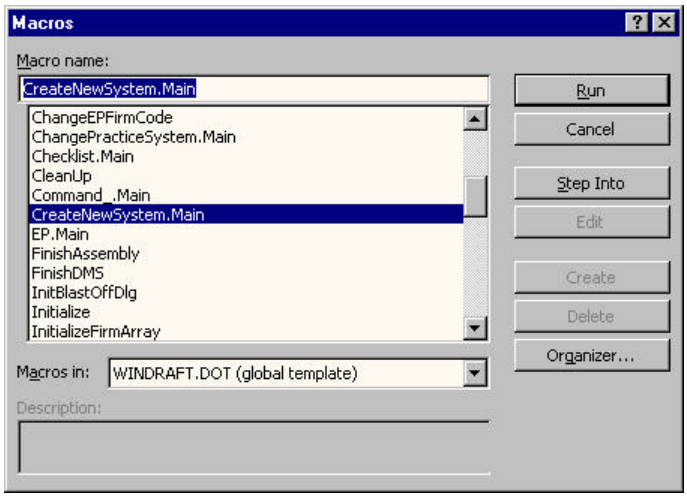
1. Know how to use Windows and Microsoft Word.
2. Have WinDraft installed on your computer, have merged some documents, and have at least briefly reviewed the *Quick Start* (Tab 1) and *WinDraft User's Guide* (Tab 2).
3. Browse through the *Author's Guide* (Tab 3) and sample Model Documents and data entry templates provided with the standard WinDraft package and any additional practice systems you have purchased.

B. Steps to Create Your Own WinDraft Practice System

1. Create the Skeleton

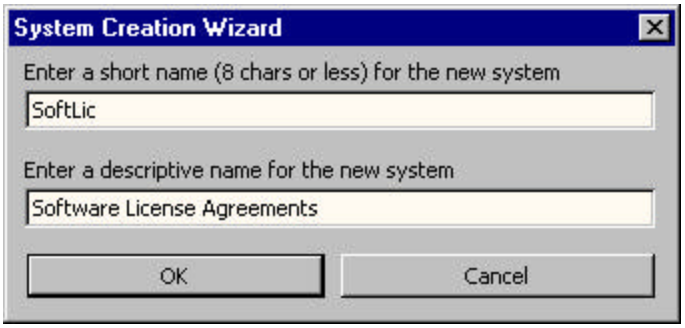
a) Run the New System Wizard macro

Run WinDraft's New System Wizard macro by selecting the Tools|Macros menu item in Word. Type "CreateNewSystem" in the text box presented, and press the Run button, like this:



WinDraft will prompt you for a short and long name for your system. The long name should be descriptive and entered in the manner you want it to appear. The short name should be some meaningful abbreviation of this, with at most 8 characters (and preferably shorter than 8).

When you have entered these names, press the OK button, and WinDraft will create a new system for you.


System Creation Wizard
Enter a short name (8 chars or less) for the new system
SoftLic
Enter a descriptive name for the new system
Software License Agreements
OK
Cancel

(This Wizard automatically creates the system directories, ".INI" file settings, etc., described in more detail in the addendum below. It will also automatically copy a sample checklist template and some sample Model Documents to a directory with the name you assigned to the Create New Practice System Wizard, and ready for you to add your own questions, variables, and text to this directory.)

You can go to the View | Options menu item from WinDraft Checklist to view and edit practice system settings.

b) Create Model Documents

Following the format of the sample documents the Wizard created and the Model Documents provided with the sample systems, use File Manager, Word, or any other method you want to copy each of the documents you want to have in your system into the \WINDRAFT*SysName* directory. Set up the beginning of each document with the [**GETDATA**] command and some [;; comments] to describe the document.

c) Create Common INCLUDE and RULES files

If you have any text which you are sure will be common between two or more of your Model Documents, create **INCLUDE** files for them in the "subdocs" directory. Similarly, if you have any long sections of logic (for gender setting routines, etc.) that you know you will need, create **RULES** (text format) files for them. Again, don't worry too much about editing these yet.

d) Create the Data Entry Outline

After testing the sample systems that came with WinDraft, you should have some idea of how you would like *your* data entry screen to look. Follow the instructions in the *Author's Guide* at Tab 3 for creating an outline to match your system. As this is just a first pass, don't worry too much about getting it perfect; you'll have to come back and edit it anyway. Also, don't create too many data entry questions yet. It will be easier to keep track of what you're doing if you add these while you are adding variables to your WinDraft documents.

e) Test

After you have added a few variables to the data entry screen:

1. Save it using the "Save Checklist Template" button on the Author Toolbar, and
2. Follow the steps in the "Quick Start" manual at Tab 1 to test your system. If you can't get it working at this point, call Eidelman Associates for assistance with troubleshooting.

2. Adding Your Language

a) Import your forms

Now is the time to bring in your documents. If you have the text in a word processing document, then insert it into the Model Documents (just cut and paste) that you already created. Otherwise, type them in.

b) Move Common Language into INCLUDE Files

Cut and paste the language which is common between two or more of your forms into the **INCLUDE** files that you already created. This way, as you update an **INCLUDE** file, all of the documents which use that **INCLUDE** file will be updated. Otherwise, you would have to make updates in multiple places.

C. Add WinDraft Logic

Making your Model Documents behave the way you'd like them to is a non-trivial task. Therefore, rather than doing the following steps all at once, you should proceed carefully, one step at a time. Program your first page, test it, then your second and test, and so on. This will make finding your mistakes much easier.

1. Identify Variables

a) Identify WinDraft variables

Read through your Model Documents and identify the places where you have text which can change from one output document to the next. Examples of this are names, amounts, and so forth. Replace these with bracketed WinDraft variables, as documented in the *WinDraft Author's Guide*.

b) Add Questions for these variables

Add questions to the checklist template so that the users of your WinDraft practice system can enter the data for the cases they are working on. Most variables of this type will probably be fill-in-the-blank (**TEXT**) type variables. Each variable in a Model Document should have a matching variable in the data entry template. However, if you have any variables for which the user will want one or more of several choices, don't hesitate to use a **CHOICE** or **CODE** question. For more information, refer to the *CheckList Author's Guide* and the Babysitting and Loan Agreement demonstration forms provided in the manual. As you work, remember that you must save the changes in the checklist template by using the button, before they will take effect.

2. Add Logic

a) Gender Logic

If you have any places where several places in the output document depend on one factor (such as the relationship between gender and pronouns), then add one **CHOICE** or **YESNO** question to your checklist template, and add IF statements in your Model Document which set variables for use in your language. Alternately, you could use the Fields in Word to handle this. See the Babysitting sample system for an example of this.

b) Identify Optional Text

There are probably several places in your Model Documents in which the practice system user might want one of two or more blocks of text in a specific place in his or her document. Follow the instructions in the *WinDraft Author's Guide* to mark off these blocks of text with appropriate **IF** ... **ENDIF** statements.

Add appropriate questions to your checklist template to allow the user to decide which block of text he or she would like incorporated into the merged document.

3. Test Your Work

Create some sample data, using the WinDraft Checklist Interface, and test the documents that you have created.

D. Prepare for Use

1. Test The Output

Work with your system for a while, and try creating some documents with it. If you find any errors, legal or otherwise, fix them. This may take some time, as WinDraft has the power to automate even very complicated documents.

2. Document Your System

Put together a three-ring binder or other form of documentation. Put copies of the data entry checklists, Model Documents, and sample completed documents in the binder. Distribute several copies, and encourage anyone who works with the system to make notes (and date them) on the hard copies. Save these hard copies as an audit trail, and as you enhance your practice system, keep the binders up to date.

3. Have Others Test Your System

Before you give your system to everyone in your department, try it on a smaller scale. Give it to one or two people in your department to use for their documents. They are likely to find errors that you might have missed.

E. Roll It Out

Distribute your finished system to everyone in your department. You can now create complex documents in a fraction of the time it previously took.

II. Addendum: Manual Steps to Creating a Practice System

A "practice system" is a system for one area of law practice to:

- ?? Gather, store and work with the facts and legal decisions.
- ?? Draft one or more documents.

These are the technical steps to follow to create a new practice system. The "Create New System Wizard" will prompt you for the long and short names of your system and perform most of these steps for you.

1. Choose a "short name" of 8 characters or less, such as "Divorce" or "RealEst."
2. Create a "document directory" for the Model Documents having that name and the standard subdirectories. We usually put them off the WinDraft directory, but you can put them anywhere. For the "Divorce" system, the directory structure looks like this:

```
c:\windraft\divorce
c:\windraft\divorce\data
c:\windraft\divorce\subdocs
c:\windraft\divorce\help
```

3. Create a "checklist template" on c:\windraft\psys\template for the checklist/data entry module to read when it starts up. There will be two files, one a Word document with the DOS file name extension ".doc," and one with the extension ".wdt," an ASCII text file created from the Word document. In our example, they would appear as follows:

```
c:\windraft\divorce\template\divorce.doc
c:\windraft\divorce\template\divorce.int
```

4. Put the Model Documents you want to "merge" on the document directory for the system. Give them 8-character names that the users will understand, such as:

```
c:\windraft\divorce\Complnt.doc
c:\windraft\divorce\Interogs.doc
c:\windraft\divorce\SepAgmnt.doc
c:\windraft\divorce\RestrOrd.doc
```

Add variable names and programming commands as described in the WinDraft Author's Guide.

5. If you use "subroutine" documents or "rules" files that are "included" in the Model Documents, they go on the "subdocs" subdirectory, like this:

```
c:\windraft\divorce\subdocs\SetHeShe.doc
c:\windraft\divorce\subdocs\PropDiv.doc
c:\windraft\divorce\subdocs\DivRules.rul
```

6. Edit WinDraft.INI to create settings for the system. (See Appendix IV: WinDraft.INI Settings for an explanation of ".INI" file settings. The Technical Appendices are in the file WDAPPNDX.DOC on the distribution disks.) In our example, two parts of WinDraft.INI would be affected: one, a section that has the same [Heading] in brackets as the short name, and (2) one line in the [System List] so that it will appear as an option when the user clicks on the change systems button. The entries might look like this:

```
...

[Divorce]
DocDir=C:\WINDRAFT\Divorce
SubDir=C:\WINDRAFT\Divorce\subdocs
WddDir=C:\WINDRAFT\divorce\data
BMacro=BlastOffGeneral
BFile=DivData.txt
ChecklistTemplate=Divorce.wdt
DefTemplate=c:\windraft\template\Divorce.dot
AltTemplate=c:\windraft\template\DivCour.dot
AltFormat=0
ResetToStyles=0

[System List]
...
...
System5=Divorce="Divorce Practice System"
...
```

Much of the work of editing the WinDraft.ini file can be handled by using the View | Options dialog in WinDraft Checklist.

7. Each practice system can have one primary and one alternate word template. When a new document is created by WinDraft, it bases the new document on this template. These two templates should be created and put in the Word template directory, with names and directory locations that match the setting in WinDraft.INI for the practice system.

I. Creating Custom Blast-off Dialogs

Easily Changing Blast-off Dialogs from Text Files

In older versions of WinDraft, custom Blastoff dialogs were supported through WordBasic programming. This method produced great results, but took a significant amount of time, and required a "real programmer." In response to this, we have created the ability for the general and estate planning dialogs to read a text file for the information that is displayed on the screen. You can edit them with Word, Notepad, or any other editor. No ini file settings are required.

Both text files are placed on the practice system directory, such as C:\windraft\EPFirmName or N:\windraft\Reloan.

New Tabbed Interface for Blastoff

Here is the new estate planning blastoff.

The first tab contains the standard estate planning documents.

Like the original one, it is very convenient for selecting mirror-image documents for husband and wife, and inapplicable documents will be grayed out.

Setting up what documents are associates with each check-box are set in the [EPXX-Documents] section of the WinDraft.ini file, and can easily be changed.

Estate Planning Blastoff's EPDocsTemplate.txt
You can change the text displayed on the estate planning tab by creating a text file called EPDocsTemplate.txt on the practice system directory.

Here is a sample:

```
"WPOA","Health Care DPOA"
"WDPOA","Property DPOA"
"WlivWill","(Reserved)"
```

The entry on the left is the WinDraft.ini [EPXX-Documents] setting for the document on that line, and the entry on the right is the text to be displayed.

General Blastoff's BlastoffTemplate.txt

For the general blastoff, (macro named BlastoffGeneral) or estate planning blastoff (BlastoffEstatePlanning), simply create a text file on the practice system directory, and name it "BlastoffTemplate.TXT".

The general blastoff, which can be used for litigation, banking, or any other application, or the general part of the estate planning interface, looks like this:

	To create a Tab, like a tab on a file folder, simply use a "NEWTAB" line, with a comma between the NEWTAB command and the label text to be displayed on the tab. For each document on a tab, enter the name of the Word document or template that gets merged when the user selects this item, and the text to be displayed to the right of the checkbox for the item. Each is in quotation marks, and separated by a comma.

The text in the BlastoffTemplate.txt file looks like this (note: the following example does not exactly correspond to the sample displayed above):

```
"NEWTAB", "Additional EP Documents"
"FirmStandardInvoice.dot","Firm Standard Invoice to Clients"
"TransferLetter.dot", "Transfer Letter"
"Blueprint.dot","Engagement Letter-Blueprint"
"NEWTAB","Corporate"
"ArticlesOfIncorporation","Articles of Incorporation"
… etc.
```

Note: Follow this syntax carefully. The program that reads this file is very particular about having the right number of spaces, quotation marks in the correct places, and so forth.

For the general blastoff, this will provide the complete interface and list of documents.

For the estate planning blastoff, this will provide additional documents to supplement the standard estate planning blast-off screen, which lists checkboxes in a column for husband and wife for standard trusts, will, and powers of attorney.

This does have a few limitations: No IF…THEN… logic is supported, no graying out of entries, and each document entry, including text that is a label, has a check box next to it. Also, no remarks are allowed.

II. WinDraft helper functions (in Word)

The following are WinDraft functions that you can use in your VBA programs to extend WinDraft. You only need to know about these if you are trying to extend WinDraft with your own Visual Basic programming.

1. INI File Routines

a) inival$=WD.GetProfile$(section$, key$)

Gets a profile value from the WinDraft.INI file. Section$ should be the desired ".INI" file section, and key$ should be the desired ".INI" file variable. This, as with all other ".INI" file functions, returns an empty string if the ".INI" file variable does not exist.

b) tempval$=WD.GetTemp$(section$, key$)

Gets a temporary value from the WinDraft.INI file. This should be used for getting PROFILE information from [internal] section of the ".INI" file. It is important that you use this for this purpose, as future versions of WinDraft will use the system registry to store profile information. At this time, GetProfile$ will use the registry, while GetTemp will use an ".INI" file or temporary file to store such information in. Section$ should be the desired ".INI" file section, and key$ should be the desired ".INI" file variable.

c) Dadir$=WD.GetDADir$(key$)

A synonym for GetDAProfile$.

d) daprofile$=WD.GetDAProfile$(key$)

Gets an ".INI" file value from the current document assembly system. Otherwise identical to GetProfile$.

e) WD.SetTemp(section$, key$, value$)

Sets a variable in the temporary file to a given value. This is the reverse of GetTemp$.

f) WD.SetProfile(section$, key$, value$)

Sets a variable in the ".INI" file to a given value. This is the reverse of GetProfile$

g) WD.ClearTemp(section$, key$)

Removes a variable for the ".INI" file. Useful for cleaning up temporary values after your program finishes.

h) trueorfalse=WD.BoolProfile(inival$)

Determines whether a text string is a "1", "TRUE", "ON", or "YES", and returns 1 if this is true.
2.

2. General Helper functions

a) WD.LookupVar(var$)

Looks up an answer in the current set of blast off data. This function is useful for seeing how the user answered the data entry questions and for graying out inapplicable documents based upon this.

b) WD.SetPracticeSystem(practicesystem$)

Sets the practice system to the given system. This should be the system's short name, which is the same as it's ".INI" file section.

c) Filename$=WD.GetFile$(code$)

Looks up a filename from the *FileSection* part of the ".INI" file for the given system. Described in more detail in the documentation on Creating BlastOff macros.

d) WD.SetAltFormat(UseAltFormat)

Determines whether to use the alternate formatting for output documents. Pass 1 to use the alternate template, 0 to use the default.

e) WD.SetWatchAssembly(UseWatchAssembly)

Determines whether to watch assembly. This function also minimizes the Word window if WatchAssembly is off.

f) WD.FinishAssembly

Should be called after document assembly is completed, but before FinishDMS is called.

g) WD.CleanUp

Should be the last thing called in a blastoff macro.

h) WD.StartDMS(SaveInDMS)

Initializes document management. The parameter passed should be 1 if the output documents are to be saved into document management, 0 otherwise.

i) WD.FinishDMS

Cleans up after document management saving, and presents a pick list of the documents just created.

j) WD.Initialize

Called to initialize WinDraft. If you wish to dynamically adjust toolbars, etc., make your adjustment in the ".INI" file, and then call this subroutine.

k) WD.InsertDoc(filename$)

Inserts a document from a file, or, if it is open, copies it from the window it is open in. Normally called by AssembleIt.

l) WD.AssembleIt(docname$)

Does the work of opening, assembling, and saving to document management. Should be called once for each document desired.

III. Using VBScript or JavaScript with WDData

The WinDraft Checklist program (WDData) supports customization on a by practice system basis through Microsoft ActiveScripting, which includes VBScript, JavaScript, as well as more obscure scripting languages such as PerlScript and Python. This allows programming custom features into WDData to meet more unusual needs.

Setting up a practice system to use scripting requires only one line in the checklist template file. Near the top of your checklist template, before the OUTLINE section, add an ACTIVESCRIPT line, with the name of your scripting language as an argument (or no argument to use VBScript). For example:

ACTIVESCRIPT
ACTIVESCRIPT VBScript

ACTIVESCRIPT JavaScript
ACTIVESCRIPT PerlScript

When you start WDData with this checklist template, the script file is loaded from the same path an filename as the wdt file, but with an extension of wds. For example, if your checklist template is C:\Program Files\WinDraft\Babysit\Babysit.wdt, then an **ACTIVESCRIPT** line will load a VBScript file from C:\Program Files\WinDraft\Babysit\Babysit.wds.

Here is an example wds file. If you use this, then WDData will ask for confirmation before blastoff. Computations could be performed before blastoff in a similar manner.

```
Sub Document_BlastOff(fCancel)
  If MsgBox("Are you sure you want to begin document assembly?", vbYesNo, "WinDraft
          Checklist") = vbNo Then
          fCancel = True
  End If
End Sub
```

To learn how to program VBScript or JavaScript, refer to Microsoft's documentation.

Events

Sub Application_Quit()
Sub Document_New()
Sub Document_Open()
Sub Document_Save(ByRef fCancel)
Sub Document_Close(ByRef fCancel)
Sub Document_BlastOff(fCancel)
Sub Document_LineLostFocus(LineName)
Sub Document_LineGotFocus(LineName)
Sub Document_ButtonClick(ButtonName)

Objects, Methods, and Properties

Application
Document
System
Variables (collection): Access by variable name (fast), integer order (slow), or For Each enumeration (fast, but high initial overhead)
Variable
Bookmarks (collection): Access by string name (slow), integer order (slow), or For Each enumeration (fast, but high initial overhead)

Bookmark

Use the VBA object browser to examine methods and properties. Start VBA, go to Tools | Reference and add a reference to "Eidelman Associates WinDraft Checklist" (click Browse… and choose the wddata executable if it's not already on the list). Go to View | Object Browser, and select the object you are interested in from the list. You can browse the properties and methods of each object, and see what parameters they take.

Error codes

13	Type mismatch.
1002	A visible page with the given name could not be found.
1003	An active line with the given name could not be found.
4248	This command is not available because no document is open.
5174	This file could not be found.
5316	WinDraft could not find or run the Microsoft Systems Info application.
5722	This property is only available if the document is an OLE object.
5723	This property is not available if the document is an OLE object.
5843	One of the values passed to this method or property is out of range
5868	Window is maximized
5869	Window is minimized
5941	The requested member of the collection does not exist.

WinDraft™

Expert Document
Drafting Software
For
Microsoft Word

Technical Appendices

Eidelman Associates

317 South Division
Suite 187
Ann Arbor, MI 48104

734-769-1500

734-769-1501 (Fax)

support@lawtech.com

www.lawtech.com

Microsoft Certified Solution Provider

WinDraft program/macros/and field programming and Documentation Copyright © 1994-2000 Eidelman Associates, all rights reserved.
No part of this publication may be stored in retrieval systems, transmitted or reproduced in any way, including but not limited to photocopy, photograph, magnetic or other record, without prior agreement and written permission of Eidelman Associates.
WinDraft and EP Expert are trademarks of Eidelman Associates. All other brand and product names are trademarks or registered trademarks of their respective companies.

WinDraft™

Technical Appendix

Table of Contents

I. Appendix: Optimizing WinDraft Performance 1
- A. Hardware Enhancements 1
- B. Using WinDraft Settings and Commands to Increase Speed 1
- C. Computer Setup Options to Enhance Performance 2

II. Appendix: Installing WinDraft 2
- A. How to Install WinDraft 2
- B. Network Setup Introduction: 3
  1. Background and issues 3
  2. Desktop Users 4
  3. Laptop Users 5
  4. Remote Users 5
  5. Alternate Test Directories 5
- C. Manual Steps for Special Situations 5
- D. Network Installation of WinDraft -- Setting up each user 6
  1. Introduction 6
  2. Typical Network Setups 7
  3. Completing the Network Installation. 7
  4. Batch Files to Copy Needed Individual Files 7
- E. Troubleshooting 8
  1. Errors while running setup. 8

III. Appendix: Component Parts of WinDraft 9
- A. WDData.exe 9
- B. _WDBO.exe 9
- C. WinDraft.wll 9
- D. WinDraft.dot 9
- E. WinDraft.ini 9
- F. WDLoad.dot 9
- G. WDFmt.dot 9
- H. WDFmtAlt.dot 9
- I. WDLSUP.BIN 9

IV. Appendix: Windraft.ini settings 10
- A. [WinDraft] 11
  1. WinDraftDir=C:\WINDRAFT 11
  2. ProgDir=C:\WINDRAFT 11

3. DefaultSystem=EP-System 11
4. Author=1 11
5. Editor=1 12
6. WatchAssembly=1 12
7. AlwaysWatch=0 12
8. UserID=*ENV(UID) 12
9. LogFile=H:\WINDRAFT.LOG 12
10. SaveDMS=DOCSOpen 13
11. SaveInDMSOn=0 13
B. [DOCSOpen] 13
1. Library=LLDOCS01 13
2. DOCSCustLegal=1 13
3. CreateUniqueDocName=1 14
4. DOCSCust*n*=Group=GROUP.GROUP_NAME 14
C. [WDData] 14
1. Path=C:\WINDRAFT\WDDATA.EXE 14
2. ChecklistTemplateDir=C:\WINDRAFT\TEMPLATE 14
3. StartCollapsed=0 14
4. HostWP=Word9 14
5. ODMA=0 14
6. DOCSOpen=0 15
D. [System List] 15
E. [New System Wizard] 16
F. [<SysName>] Follow this format for each practice system (see examples below) 17
1. DocDir=C:\WINDRAFT*SysName* 17
2. SubDir=C:\WINDRAFT*SysName*\subdocs 17
3. FileSection=File Section Heading Name 17
4. WddDir=c:\windraft*SysName*\data 17
5. BMacro=BlastOffGeneral 17
6. Bfile=SNamData.txt 17
7. ChecklistTemplate=C:\WinDraft*SysName*\Template*SysName*.wdt 18
8. ChecklistHelp=C:\WinDraft*SysName*\Template*SysName*.nfo 18
9. ChecklistHelpURL=http://www.lawtech.com/ephelp/ 18
10. ChecklistHelpURLFormat=%b%p.html#%l 18
11. DOCSQBE=TYPE_ID='*SysNameData*' 19
12. DefTemplate=C:\WINWORD\TEMPLATE\wdfmt.dot 19
13. AltTemplate=C:\WINWORD\TEMPLATE\wdfmtalt.dot 19
14. AltFormat=0 19
15. ResetToStyles=0 19
G. [internal] 20
H. [WDData Preferences] 20
I. [EP-System] 20
J. [EP-Documents] 20
K. [ABALoan-System] -- Example for ABA Shootout loan system 20
L. [ABALoan-Documents] 20

M. [Babysit-System] 21
N. Sample ini file settings 22

V. Appendix: Integration with DOCS OPEN Document Management 25

A. Introduction 25
1. Basic Compatibility 25
2. Saving and Opening Data Files 25
3. Automatic Document Profile Creation 25
B. The User's Perspective 25
1. Data Files in DOCS OPEN 25
2. Automatic Profile Creation 26
C. Technical Notes 26
1. The WinDraft.INI file 27
2. WDData 28
3. Automatic Creation of Profiles 29
D. Troubleshooting 31
1. Working with WDData Data Files 31
2. Output to DOCS OPEN 31
3. Problems Returning "Hit List" of Documents Created 33

VI. Appendix -- Troubleshooting 34

A. Macro Button Doesn't Work Right 34

VII. Appendix -- Model Document Structure for EP Expert 34

A. LONG FORM WILL 34
1. WillHusb.Doc or WillWife.Doc 34
B. REVOCABLE TRUST 35
1. RT_Husb.Doc or RT_Wife.Doc 35
2. GRGender.Doc 36
3. Trust.Doc 36
C. IRREVOCABLE TRUST 36
1. IRTrust.Doc 36

WinDraft Technical Appendices

I. Appendix: Optimizing WinDraft Performance

In the early days of WinDraft, we created a Revocable Trust with several hundred IF statements and almost 200 pages of text that might or might not be included. WinDraft merges it in under a minute on a Pentium 90 machine with fast video card and an optimized setup. It can take as long as 15 minutes on a 386 that isn't optimized.

Today's machines are much faster, and usually come from the factory reasonably well-optimized. Nonetheless, there are a number of things that can be done to enhance WinDraft performance further.

A. Hardware Enhancements

Any enhancements to hardware that make Word run faster will also make WinDraft run faster. The most significant are:

- Faster processor
- Video accelerator with matching drivers--very important.
- More RAM (at least 32, preferably 64 MB or more)
- Fast hard disk and hard disk controller. WinDraft reads and writes files, and Word creates a lot of temporary files.
- Fast network cards, if running from a server.

B. Using WinDraft Settings and Commands to Increase Speed

WinDraft has two ways to turn the video display off while the document is being assembled: (1) the model document can contain DISPLAY ON and DISPLAY OFF commands; and (2) the user can click the "Watch Assembly" check box in the "blast off" dialog box to turn off the display of the document being assembled. The user sees a status window that displays messages and status information, but doesn't see the actual document being processed. Turning off the display will increase performance by as much as 15%, even on a machine with fast video.

There are two other ways you can speed things up significantly that involve programming techniques.

First, if you have long subroutine documents that contain IF ... SET commands, such as those we use in the trust agreements to set trustee names and the gender terms of the parties, you can cut the time it takes to process these commands

dramatically by using the RULES command instead of INCLUDE. (See *Author's Guide).*

Second, if you have a long section of text that is only used under certain situations in your main model document, you can move it to a "subroutine document," and INCLUDE it when you need it. This way, WinDraft doesn't have to crunch through that text unless the text is needed.

C. Computer Setup Options to Enhance Performance

- Run Word/WinDraft from local hard drive rather than from a network drive.
- Don't use Stacker or Doublespace disk compression, at least for files that are used a lot by WinDraft. (*e.g*. C:\temp directory.)
- Make sure the right drivers are loaded for high speed video cards. With ATI cards, we found that switching from SVGA to ATI drivers increased performance by as much as 20% when "watch assembly" is set to "on," and 10% when "watch assembly" is "off."

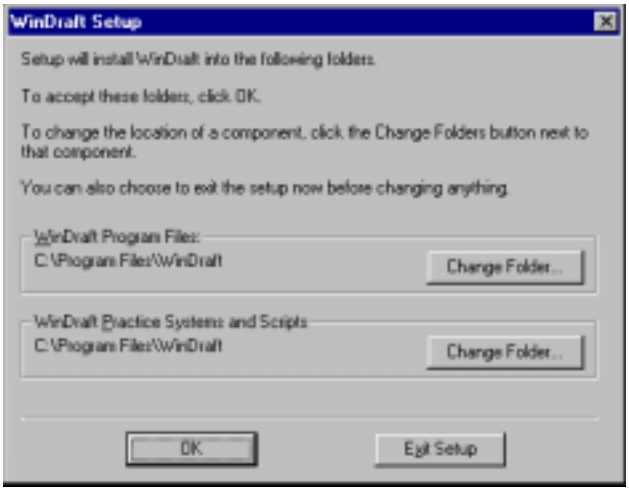
II. Appendix: Installing WinDraft

A. How to Install WinDraft

WinDraft is typically shipped with the programs and sample practice systems on floppy disk. The following is a description of the installation process for a disk that contains the programs and sample model documents provided with EP Expert, the estate planning system. It will vary slightly if you are using a different demo or model document system.

Insert the WinDraft diskette in Drive A:. Exit from WinWord (and WDDATA, if you already have it running). Go to the Start menu and select Run. At this dialog box, type "**A:SETUP**", and click on OK.

The setup program will first give a welcome message. Click ok. The following dialog will then come up:


WinDraft Setup
Setup will install WinDraft into the following folders.
To accept these folders, click OK.
To change the location of a component, click the Change Folders button next to that component.
You can also choose to exit the setup now before changing anything.
WinDraft Program Files:
C:\Program Files\WinDraft
Change Folder...
WinDraft Practice Systems and Scripts
C:\Program Files\WinDraft
Change Folder...
OK
Exit Setup

For a standard single-user setup, we suggest going with the default. That is, put WinDraft on C:\Program Files\WinDraft directory.

For network setups, make sure you are logged in with a supervisor's ID. The default will be to install everything on the C drive. Most network setups keep the WinDraft Practice System and Script on the network drive and WinDraft program files on the local drive. See "Network Setup Notes" below for a discussion of how to tweak where various files go.

After you click the "Continue" button, you will see each step as setup does its work.

The SETUP utility works much like the Microsoft Word setup utility and will perform the following steps:

- Create the necessary WinDraft directory and subdirectories.
- Uncompress and copy most WinDraft system and model document files to the WinDraft directory and subdirectories.
- Uncompress and copy the WDLoad.DOT global add-in to the WinWord startup directory.
- Uncompress and copy formatting and style templates (wdfmt.dot, wdfmtalt.dot).
- Create WinDraft.ini, with appropriate settings, in the Windows directory.

B. Network Setup Introduction:

1. Background and issues

The various components of WinDraft can be located on any hard drives the PC can access.

The default is to install everything except the WinDraft.ini and a couple of Word templates on c:\windraft, with further subdirectories for each practice system, and further subdirectories for subdocs, data, and htmlhelp, like this:

C:\windraft\
C:\windraft\litigation
C:\windraft\litigation\data
C:\windraft\litigation\subdocs
C:\windraft\litigation\HTMLhelp

If you want, you can replace the "C" above with any drive letter you want, and for that matter, you can use other directory names, as well.

2. Desktop Users

In general, we follow the network philosophy that, given that there are no performance problems, it is generally better to have both hardware and software on the network. This makes it easier to update, manage, support and backup.

If you copy things do the local drive, they run faster, but the files need to be updated each time there is a change.

At the beginning, we usually install to Drive C:, just to get people started, without having to worry about any of this.

Type of files	10 mb network	100 mb network	Notes
Data	Network	Network	Data files are quite small
Model documents	Network (but it is noticeably slower, so some move to local drive to optimize, especially with extremely big practice system	Network	
WinDraft programs	Depends on version: Word 95: Network Word 97: Local	Network	In the 97 version, which uses VBA, WinDraft.DOT ballooned and is noticeably slow to load (just the first time).
HTML Help	Network	Network	Same as an intranet.
Checklist Templates	Network (unless you have huge checklist	Network	

	templates)		

3. Laptop Users

If they are going to be running off-line, set up everything locally, but with easy synchronization when on the network. If there is document management, turn on regular document management when on the network.

Set up procedures to synchronize, backup, archive, etc.

4. Remote Users

a) Using RAS or VPN:

Everything on the local drive, except, possibly, for data files and HTML help.

b) Using Citrix, PC Anywhere, etc. for remote control:

Everything is on the network.

5. Alternate Test Directories

If files are being updated on a network, it is generally best to test them on one directory, then once they are working right, roll them out to the workgroup that is sharing the production directory.

If this can't be done, then testing on Drive C: will work fine for that purpose.

C. Manual Steps for Special Situations

Simply clicking the mouse on "OK" will work fine for a typical setup on a stand-alone system. However, you may need to make some manual changes in special situations. Feel free to call Eidelman Associates at 313-769-1500 for help.

WinDraft is an "open system" that can be configured in a wide variety of ways.

Some special situations include:

- Network setups -- (See next section for instructions on completion of network setup for each user or workstation.)

  The model documents, the WinDraft engine and various templates can be called from a network drive or your local hard disk. This is true whether you are running Windows or WinWord from the network or locally. As long as the

WinDraft.ini file points to valid directories and the files are copied there, you can locate any of the files anywhere you want.

- Alternate way to load WinDraft.

  By default, the WinDraft global template files will be installed in the WinDraft directory, to be loaded and unloaded by pressing Control-Alt-W. If you wish, you can copy WinDraft.DOT and WinDraft.WLL to the \Winword\startup directory so they will automatically load every time you run Word. To do this:

  1. Copy WinDraft.DOT and WinDraft.WLL to the WinWord startup directory from \WinDraft.
  2. Edit WinDraft.ini so that ProgDir=C:\Winword\startup (or similar directory for your system.)

  Note that Control-Alt-W will still load and unload WinDraft. The only difference is that WinDraft will be loaded at the beginning of every Word session.

- Use with Document Management Systems

  WinDraft has special commands and functions to interface with document management systems, such as DOCS OPEN. Settings in WinDraft.ini and the DOCS OPEN tables can be set up so that:

  3. WinDraft's data entry module can use DOCS OPEN to manage client data files.
  4. WinDraft can automatically save each document it creates in DOCS OPEN and automatically fill out each field of the profile.

  See Appendix, DOCS OPEN Integration, for further information.

  WinDraft also supports any ODMA compliant document management system for storing the information entered in the checklist program. To enable this, see the ini file documentation.

- Integration with Databases

WinDraft can automatically accept data from Microsoft Access and other databases and integrate with databases in a number of ways. In each case, installation will be customized, depending on your database setup.

D. Network Installation of WinDraft -- Setting up each user

1. Introduction

Law firms and law departments are set up in a wide variety of ways. Some have just about everything loaded locally, on drive C:, and others have just about everything loaded on the network.

For purposes of discussion, we will make the following assumptions. You will have to make whatever adjustments are appropriate for your situation.

Local Drive:	C:	(Some use D:)
Shared Network Drive:	F:	(Yours could be anything)
Windows installed on:	\WINDOWS	
Microsoft Word installed on:	\MSOFFICE\WINWORD	

2. Typical Network Setups

The most common setup our new clients already have up and running is Microsoft Word and Microsoft Windows running locally, on the C: drive, with data and firm-wide applications running on the LAN.

Some have Word running from the LAN, with Windows loaded locally, and some have everything running from the LAN server.

With any of these setups, we recommend installing WinDraft to your LAN server. That is, when running the setup program (A:SETUP), at the prompt type "F:\WINDRAFT" for the WinDraft installation directory. All of the WinDraft files except those noted below will then be set up for shared network use.

SETUP also installs a few user-specific files or Windows system files that must be individually set up for each WinDraft user. SETUP will automatically set these up for the PC on which SETUP was run, so the manual steps should be done for each other user.

3. Completing the Network Installation.

SETUP installs the following files to the following locations:

FILE	TYPICAL LOCATION	EXPLANATION FOR NON-STANDARD INSTALLATIONS
WinDraft.ini	C:\windows	Each user has his or her own Windraft.ini file, located on the directory from which Windows is loaded. On some networks, this may be a "home" directory on the file server, such as H:\WINDOWS.
WDLoad.DOT	C:\MSOFFICE\WINWORD\ STARTUP	Must be on the user's WinWord startup directory.

4. Batch Files to Copy Needed Individual Files

The WinDraft setup disk also contains a sample batch file called WDNETSET.BAT that is set up to copy these files from a directory on the network called \WINDRAFT\WDNETSET to the individual's hard drive or other individual file locations. You will need to first:

1. Install WinDraft to one machine and make sure it is working right. Call Eidelman Associates, if necessary, to make any changes to the setup that you need to make for your firm, such as changing file settings for document management, etc.
2. Copy the files named above from the workstation on which you have your prototype system installed to the network. (For purposes of discussion, "F:\WINDRAFT\WDNETSET".)
3. Modify the batch file so that it copies to and from the right locations and put it on the network in F:\WINDRAFT\NETSET or some other location where any user's PC can access it.
4. From each user's individual workstation, run the batch file from the network. It will copy the files that user needs.

Here is the text of the batch file for a typical configuration.

```
REM WDNETSET.BAT -- to complete WinDraft network setup for indiv. PCs.

F:
cd \windraft\wdnetset

copy  windraft.ini  c:\windows
copy  wdload.dot  c:\msoffice\winword\startup
```

E. Troubleshooting

1. Errors while running setup.

The following is a discussion of errors you may encounter while running SETUP:

a) "Access Denied"

Access Denied" -- If you already have a file open that WinDraft is trying to create, such as a global template or WDDATA.EXE, the setup program will try to copy over it and will get a "file in use" or "access denied" error. Please exit from WinWord and run setup again. (If this is not your first installation of WinDraft, and you have WDData open, please exit from that program before running setup.)

You could also get this kind of error if you are trying to install to a directory on a network to which you do not have access. If so, please contact your network supervisor or login with an ID that has access to the directories you are attempting to use or create.

III. Appendix: Component Parts of WinDraft

A. WDData.exe

The checklist data entry program. WDData should appear in the same path as the path ini file setting in the [WDData] section. (Programmed in Microsoft C++)

B. _WDBO.exe

This file is used internally by WDData to handle blastoffs to Word 2000. It should not be run directly.

C. WinDraft.wll

The WinDraft Word add in library (WLL). This part of it is what does the actual document assembly. WinDraft.wll should go in the directory pointed to by the ProgDir ini file setting in the [WinDraft] section. (Programmed in Microsoft C)

D. WinDraft.dot

The WinDraft Word template. This template contains support macros for WinDraft.wll which handle document selection, much of the DOCS OPEN integration, and so forth. WinDraft.dot should go in the directory pointed to by the ProgDir ini file setting in the [WinDraft] section. (Microsoft Visual Basic for Applications)

E. WinDraft.ini

The WinDraft "ini" file, this text file contains information about the settings of WinDraft in the standard Windows ini file format. WinDraft.ini should be located in the Windows directory.

F. WDLoad.dot

The WDLoad.dot Word template contains macros which support loading the WinDraft add-in to Word. This template must go in the \WINWORD\STARTUP directory. If this template is not present, then CTRL-ALT-W will not load WinDraft, and WDData will not be able to blast off.

G. WDFmt.dot

H. WDFmtAlt.dot

The Word templates which have the standard and alternate formatting (respectively) for WinDraft output documents. These files should go in the TEMPLATE directory under each practice system that uses them.

I. WDLSUP.BIN

Supplementary program data and licensing information for WinDraft and WDData. Should go in the directory pointed to by the WinDraftDir ini file setting in the [WinDraft] section.

IV. Appendix: Windraft.ini settings

Each user has his or her own WinDraft.ini file, a standard ASCII text file, in the standard Windows ini file format.

WinDraft.ini contains the settings that tell WinDraft:

- Where to find documents and other files it needs
- What kind of setup each user has for Windows, Word and WinDraft
- Which features to use
- How to integrate with document management
- Specific information for each practice system

In normal operation, the user does not need to be concerned with WinDraft.ini. It will be set up automatically when WinDraft is first loaded by the SETUP program and will be updated automatically by the Create New System Wizard, and by changes made under View | Options and other functions of the program.

"Techies" will need to be concerned with modifications to WinDraft.ini for non-standard network setups, for custom changes to the setup, and for troubleshooting if there are problems.

If there is a problem with the way WinDraft is working, it can usually be solved by an adjustment to the WinDraft.ini file.

Location of WinDraft.ini: WinDraft.ini will normally be located on the "Windows" directory along with most other Windows .ini files. On a network, this should be either the local hard disk (Drive C:) or the user's home directory (usually Drive H:), because each user must have his or her own WinDraft.ini file.

In large organizations, the MIS Department will control and maintain the .ini files, and may update them using a special program for batch modifications to .ini files. If you are unfamiliar with editing ini files, see your Windows documentation. You can edit an .ini file with Notepad or with Microsoft Word. (If you use Word, be sure to Save As a text file.) You can also use one of the many .ini file editors available on CompuServe and various BBS systems. Like other .ini files, the information in them can be in any order, as long as the individual lines appear in the right "section," following the defined [Heading]. In general, the settings are not case sensitive, so you can use any combination of capital and lower case letters.

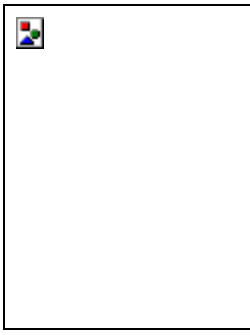
Be sure to see the "readme.txt" file for additional information. The WinDraft.ini file settings may change as new features or integration with other programs are added.

The following samples and explanations describe the typical .ini file settings for a user who has access to a number of practice systems. In individual cases, your settings may be more limited or different. Call Eidelman Associates at 313-769-1500 for more information, if you need it.

As you are studying the .ini file is set-up, we suggest that you print your WinDraft.ini file out and have it on hand as you read the following descriptions of the settings.

A. [WinDraft]

Global settings which affect everything that WinDraft does.



1. WinDraftDir=C:\WINDRAFT

This is the main 'root' directory of WinDraft, usually C:\WinDraft or C:\Program Files\WinDraft on a standalone installation. This is used to, among other things, determine where a system created with the system wizard will go, where WDLSup.dat should go, and where the help file should go.

(When running SETUP from the installation disk, this directory will be the one directory for which a dialogue box will be displayed, and the rest will automatically be created based on this.)

2. ProgDir=C:\WINDRAFT

This is the directory in which WinDraft.DOT, and WinDraft.WLL, are located. It is used in loading the WinDraft environment from within Word, in loading WDData, and so forth. If it is desired to have WinDraft load when Word starts up, this should go in the Word startup directory, which is C:\WINWORD\STARTUP on a standalone installation.

3. DefaultSystem=EP-System

If this setting is present, then each time WinDraft is loaded, it will reset the Practice System to this setting. The line above, for example, would choose the Estate Planning system for the user each time WinDraft is loaded. If this setting is not present, then WinDraft uses the last selected system each time it starts up.

4. Author=1

If this setting is "1", then WinDraft knows that this user is an "author," that is, someone who writes their own Model documents and checklist templates. Currently, this setting determines whether the WinDraft Author toolbar is displayed. The typical user does not need to be distracted by looking at the Author Toolbar, which contains buttons for editing, testing and saving model documents.

5. Editor=1

Sets preference for whether WinDraft Edit Tools toolbar will be displayed when WinDraft is loaded. If this setting is "1", then the Edit Tools toolbars is displayed.

6. WatchAssembly=1

This setting is the *default* for the Watch Assembly checkbox in the blast-off dialogs. This setting should probably be 0 for most experienced users running well-tested model documents. This is because Watch Assembly is mainly useful for Authors troubleshooting problems in their documents or for new users to watch how WinDraft works. WinDraft runs about 15 to 20 times faster with Watch Assembly *off*. This setting is adjustable via Tools|WinDraft Options.

7. AlwaysWatch=0

AlwaysWatch should be set to 0 for most users, as it is only used by authors during the process of debugging model documents. It overrides the DISPLAY OFF command in a model document, forcing the display of even parts of the document which are normally hidden (Gender setting subroutines, etc.) by the "DISPLAY OFF" command in the model document. Turning this setting on slows down document assembly by about 10%. WatchAssembly must be on for this to take effect.

8. UserID=*ENV(UID)

This tells WinDraft what the user's network login ID is. This information is optionally used in the document management integration and naming of log files.

If you want to "hard code" the user's ID, the format is "UserID=USERNAME".

If you want to have WinDraft pull the user's name from a DOS environment variable, use the format above and put the name of the environment variable in the parentheses. For example, if your network is set up so that you use NUSERID as your environment variable, then this line would say "UserID=*ENV(NUSERID)".

If you are using DOCS Open, and would like to have WinDraft pull the user's name from here, the ini line should read "UserID=*DOCS". This feature requires DOCS Open version 2.0c or later.

9. LogFile=H:\WINDRAFT.LOG

If this setting is present, WinDraft opens this file and logs the start and finish of document assembly, as well as extra information specified in each document, into the given file using the WinDraft SYSLOG

command. This file should be on a local drive or, if WinDraft is being used on a network, it can be put on the user's home directory. WinDraft needs exclusive access to write this file.

In order to permit a network to be set up with each user having his or her own log file on the same directory, the setting should be in the form "LogFile=I:\Windraft\Logfiles*", where the path is the directory path where the log files are to be stored and the * indicates that WinDraft will use the UserID name for the log file. Thus, if the user's login is JONESRA, the log file will be JONESRA.LOG on the indicated directory.

10. SaveDMS=DOCSOpen

Determines whether WinDraft should attempt to automatically save and profile documents that WinDraft generates into DOCS Open or LawOffice. Leave this setting blank or omit for systems without a document management system or that do not use automatic profiling. Set to DOCSOpen or LawOffice to enable the 'Save in document management' checkbox.

Users may still decide whether to save a particular run to document management or not.

11. SaveInDMSOn=0

This setting determines the default value (checked or unchecked) of the 'Save in document management' checkbox. It normally should be set to 0.

B. [DOCSOpen]

1. Library=LLDOCS01

This setting determines which DOCS Open library, if any, the WinDraft DOCS Open integration attempts to use to save automatically profiled documents to. This is used in the automatic DOCS Open profiling.

If this setting is *DOCS (or not present), then WinDraft will get the current default library from DOCS Open. This feature requires DOCS Open version 2.0c or later.

2. DOCSCustLegal=1

This setting should be set to "1" if your organization is using the Legal sample customization supplied with DOCS Open. If you specified your industry as "Legal" during DOCS Open setup, and you have Client and Matter fields on your profile form, then you should set this to "1". Otherwise, it may be omitted.

3. CreateUniqueDocName=1

This setting should be "1" if you are using DOCS Open 2.5 or later with unique document names. This setting may also be enabled with earlier versions of DOCS Open, but versions earlier than 2.5 support only 30 character docnames. Since the date and time stamp that is added to the end of the docname is about 20 characters by itself, this leaves little space for a meaningful document name.

4. DOCSCust*n*=Group=GROUP.GROUP_NAME

The settings DOCSCust1 through DOCSCust9 are used for supporting additional profile fields on the DOCS Open profile. For example, if you have added a Group field to your profile form, which stores the possible groups in a Group_Name field in a Group table, you might add the line "DOCSCust1=Group=Group.Group_Name" to your ini file.

Contact Eidelman Associates for more information on setting up custom DOCS Open fields in WinDraft.

C. [WDData]

Settings affecting the WinDraft Data Entry program, WDData.exe.

1. Path=C:\WINDRAFT\WDDATA.EXE

The full path of where WinDraft can find WDData. This is used in the Checklist button for loading WDData.

2. ChecklistTemplateDir=C:\WINDRAFT\TEMPLATE

The path where WDData looks to find "scripts" telling it what questions to display. This is used in loading WDData. WDData tries to find the script specified in the practice system section in here if a full path is not specified.

3. StartCollapsed=0

If this is set to 0, WDData starts with the checklist outline fully expanded. If it is set to 1, then WDData starts with only the first level of the outline visible.

4. HostWP=Word9

The HostWP setting controls which word processor WDData attempts to use for document assembly when the blastoff button is clicked. Set this via View | Options. Supported are WordPerfect (WP), Microsoft Word up to v8 (Word), Microsoft Word v9 (Word9), and none.

5. ODMA=0

Setting is used to optionally make manual adjustments to WinDraft's automatic compatibility with ODMA. Compliant document management

software, such as iManage, SoftSolutions, Saros and DOCS Open. When WDData opens and saves files and this setting is not present, WinDraft will automatically use the normal Windows dialogs if there is not an ODMA compatible document management system loaded, and will use the document management system if it is present.

If ODMA=1, WinDraft will display an error message if, for some reason, the document manager is not loaded. If ODMA=0, WinDraft will use the normal Windows dialogs (even if an ODMA document manager is loaded).

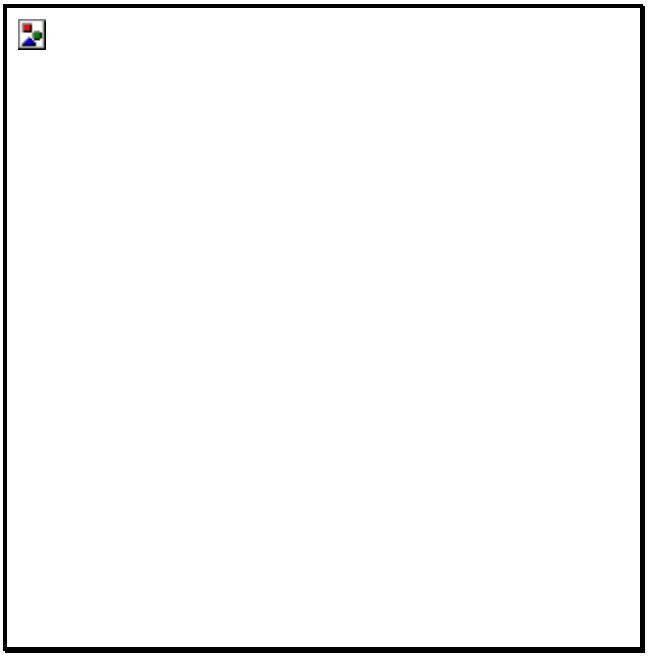
6. DOCSOpen=0

Determines whether WDData uses DOCS Open for File|Open and File|Save. If this setting is on, then WDData pops up the standard DOCS Open dialog boxes for these file functions, otherwise, WDData uses the standard Windows dialog boxes. A shift key may be used to override the DOCS Open integration, as is standard.

You should set this to 1 only if you have problems integrating WDData with DOCS OPEN through ODMA. Eidelman Associates no longer supports the native DOCS OPEN support, and includes it for backwards compatibility only.

D. [System List]

System1=EP-System="Estate Planning Documents"
System2=-Corp-System="Corporate Documents"
System3=Release-System="Releases"
System4=-Software-System="Software License Agreement"
System5=Divorce-System="Divorce Practice System"
System6=-RE-System-System="Real Estate Documents"
System7=ABALoan-System="Loan Agreements"
System8=Response-System="Response to Audit Request"
System9=Market-System="Marketing"



When you click on the "Change Systems" button, WinDraft displays a dialogue box with a button for each practice system. Those you have access to will be in black, and those that you can see, but are not accessible, are "grayed out." This dialogue comes from the settings in this part of WinDraft.ini.

The settings that are grayed out are those with a hyphen ("-") in front of them in the .ini file.

There are settings determining what systems WinDraft knows about. There can be up to 10 systems here, corresponding with settings System1 through System10. Each of these should be followed by a short name, which is the same as the .ini file section used for the practice system, and a long name, which is what is displayed by the radio button for this system in the Change Practice Systems dialog.

In a firm where a number of practice systems are supported on a network, it is recommended that all of the firm's systems be listed, but those for other departments which the user may not access would be grayed out.

E. [New System Wizard]

IntDoc=C:\WINDRAFT\SAMPINT.DOC
ModelDoc=C:\WINDRAFT\SAMPDOC.DOC

Settings determine the files that WinDraft's Create New System Wizard uses as a "seed" for a new system. These should both use full pathnames. The IntDoc setting determines the Word document which is used for a WDData script for the new system, and the ModelDoc setting determines the document used as a starter model document for the new system. WinDraft ships with a default for these, which provides an example of WinDraft and WDData syntax, helpful in getting a new practice system started. The Wizard will copy these into your new system.

F. [<SysName>] Follow this format for each practice system (see examples below)

1. DocDir=C:\WINDRAFT*SysName*

Determines where WinDraft looks for the main documents in this practice system. This is used in the blast off dialog box routines.

2. SubDir=C:\WINDRAFT*SysName*\subdocs

Determines where WinDraft looks for subdocuments in this practice system. This is used when WinDraft finds an *[INCLUDE]* or *[RULES]* statement in one of its model document.

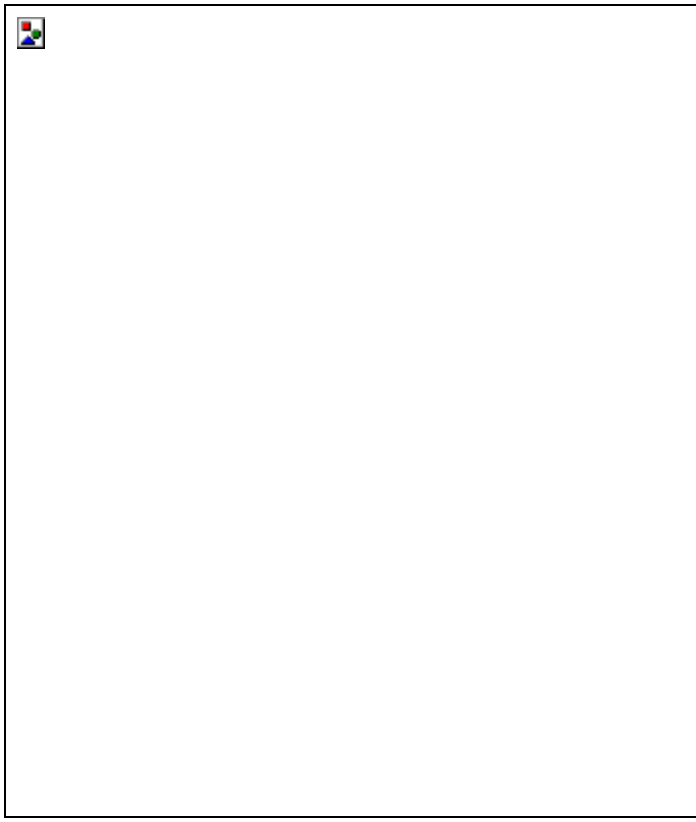
3. FileSection=File Section Heading Name

If you have a custom blast-off dialogue, it will contain a list of document names which need to be put in their own section of WinDraft.ini. This setting tells the custom blast-off where to look for these files. (See example in Estate Planning and ABA Loan systems below.)

4. WddDir=c:\windraft*SysName*\data

This setting determines where WDData by default looks for data documents for this practice system. This setting is not meaningful when DOCS Open integration is turned on.

5. BMacro=BlastOffGeneral



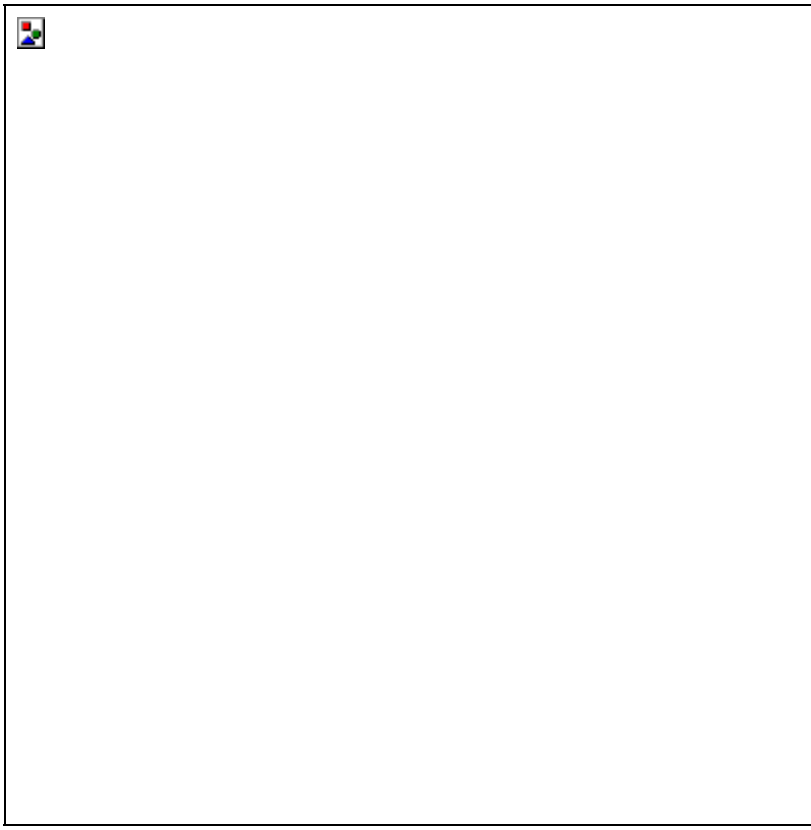
The "Blast off macro" for this system. This is the name of the macro that runs when the user clicks on the "blast off" button. This should normally be set to BlastOffGeneral, unless you have a custom blast off dialog box for this system, in which case it should be set to the appropriate dialog. (i.e. BlastOffABALoan, BlastOffEstatePlanning, etc.).

6. Bfile=SNamData.txt

The file in which WDData stores information to pass to WinDraft's document assembly. This file is created in the directory determined by the TEMP environment variable, or if none exists, on C:\. This filed is used

only internally by WinDraft.

7. **ChecklistTemplate=C:\WinDraft*SysName*\Template*SysName*.wdt**



Determines which data entry screen template (checklist template) WDData loads for this system. This determines what questions are asked in WDData. Each practice system normally has its own checklist template, an ASCII file with the extension ".WDT".

The author normally works with the checklist template as a Word document, with the name SysName.DOC, then clicks the button to automatically Save As a text file with the WDT extension.

8. **ChecklistHelp=C:\WinDraft*SysName*\Template*SysName*.nfo**

Determines the help screens that WDData uses to display Help | More Information on this system. This may included information itself, or point to information on the world wide web.

9. **ChecklistHelpURL=http://www.lawtech.com/ephelp/**

A base URL for displaying Help | More Information. Combines with information from the .NFO file to bring up context sensitive help from the web for the practice system.

10. **ChecklistHelpURLFormat=%b%p.html#%l**

A default format string to determine the URL accessed by Help | More Information. It is expanded as follows: %B becomes the base URL set in CheckListhelpURL, %P becomes the name of the current page, %L becomes the name of the current line.

11. DOCSQBE=TYPE_ID='*SysNameData*'

This setting is provided for backwards compatibility: it does not work under ODMA, and is no longer supported by Eidelman Associates.

If present, supplies Query By Example search criteria for DOCS Open to bring up a list of WDData documents in the File|Open in WDData. Only has an effect when DOCS Open integration is turned on in WDData. For example, to restrict the search to WDData documents of type EPDATA, the line used would be "DOCSQBE=TYPE_ID='EPDATA'; APP_ID='WDDATA'

Refer to the DOCS Open documentation for more information on writing a query for this line.

12. DefTemplate=C:\WINWORD\TEMPLATE\wdfmt.dot

The default Word template for WinDraft-created documents. This normally uses a "modern" font with serifs, such as 12 point Times New Roman. It should also contain styles to describe formatting of outline headings and body level styles to suit the needs of the type of documents created. See "Advanced Word Processing - Functions" for more information.

13. AltTemplate=C:\WINWORD\TEMPLATE\wdfmtalt.dot

The alternate Word template for WinDraft-created documents. This normally uses a "typewriter" font, such as 12 point Courier New.

14. AltFormat=0

Determines whether the DefTemplate or AltTemplate setting is the default on the blast-off dialogue. If this is "0," then the template named in the DefTemplate is used. If it is "1," then the template named in the AltTemplate is used. This can always be overridden by the "Use Alternate Formatting" checkbox in the blast off dialog boxes.

15. ResetToStyles=0

If this setting is "1," then WinDraft resets all of the text in the output document back to the Word style it uses. This is the same as selecting all the text in the document and pressing Alt-spacebar. This can be useful for large documents, which have many different parts and which are worked on by many different people, each of whom format it differently. However, this destroys formatting in the document which does not use Word styles. A finished, finely-tuned practice system will normally have this set to "1," with all of the needed styles set up in the templates. While converting manually-formatted documents, it is best to have it set to "0" so that the formatting of the original documents will be maintained. The default for this setting (if it is not present) is "0."

G. [internal]

Internal settings for WinDraft. These should not be edited by humans. WinDraft uses this for saving some temporary settings, as well as for document management profiling.

H. [WDData Preferences]

Preferences automatically maintained by WDData such as size and position of its window, number of pages shown in print preview, most recently used files, and so forth.

I. [EP-System]

DocDir=C:\WINDRAFT\EP
SubDir=C:\WINDRAFT\EP\subdocs
…

Standard settings for the Estate Planning Practice System. This is described in more detail under **[<SysName>]** above.

J. [EP-Documents]

HRT=RT_Husb.doc
WRT=RT_Wife.doc
HWill=WillHusb.doc
WWill=WillWife.doc
HIR=IT_Husb.doc
WIR=IT_Wife.doc
HPOA=HCPOA_H.doc
WPOA=HCPOA_W.doc
BILL=EPBill.doc

Customized part of estate planning practice system. Determines the filenames that the Estate Planning custom blastoff uses to find its model documents. These correspond with the document checkboxes in this blastoff dialog.

K. [ABALoan-System] -- Example for ABA Shootout loan system

DocDir=C:\WINDRAFT\ABALOAN
SubDir=C:\WINDRAFT\ABALOAN\subdocs
...

Standard settings for the ABALoan Practice System. This is described in more detail under **[<SysName>]** above.

L. [ABALoan-Documents]

PromNote=PromNote.doc
LSAgree=loanagt.doc
UCC9=UCC9.doc

CoverLet=CoverLet.doc
AssLifeIns=LifeIns.doc
Mortgage=Mortgage.doc
OTHD1=Other1.doc
OTHD2=Other2.doc
OTHD3=Other3.doc

Customized setup for the ABA Loan System created for the ABA TechShow Document Assembly Shootout, one of the sample systems loaded by the WinDraft setup program. This determines the filenames that the ABALoan custom blastoff uses to find it's model documents. These correspond with the document checkboxes in this blastoff dialog.

M. [Babysit-System]

DocDir=C:\WINDRAFT\babysit
SubDir=C:\WINDRAFT\babysit\subdocs
…

Standard settings for the Babysitting Demo Practice System. This is described in more detail under **[<SysName>]** above.

N. Sample ini file settings

Setting	Standalone Setting	Network Setting	Network and DOCS Open Setting	Brief Explanation
[WinDraft]				
WinDraftDir	C:\WINDRAFT	V:\WINDRAFT	V:\WINDRAFT	Main WinDraft Directory
ProgDir	C:\WINDRAFT	V:\WINDRAFT	V:\WINDRAFT	WinDraft Programs directory
DefaultSystem	Name of a system or not present	Name of a system or not present	Name of a system or not present	Default practice system on load of WinDraft
Author	0 or not present	0 or not present	0 or not present	Whether the user is an author of model documents
WatchAssembly	0 or not present	0 or not present	0 or not present	Whether the document is displayed as it is being created.
AlwaysWatch	0 or not present	0 or not present	0 or not present	Whether *all* parts of the document are displayed when WatchAssembly is used
SaveDMS	blank or not present	blank or LawOffice	DOCSOpen	Whether WinDraft can automatically saves output documents into document management
LogFile	C:\WINDRAFT\WINDRAFT.LOG or not present	C:\TEMP\WINDRAFT.LOG or not present	C:\TEMP\WINDRAFT.LOG or not present	The full path of a file used by WinDraft for logging purposes
UserID	STANDALONE or not present	*ENV(UID) or not present	*DOCS or *ENV(UID)	The ID of the current user
[DOCSOpen]				
Library	not present	not present	GPDOCS01 (or other library name) or *DOCS	The library used by WinDraft to save documents to if DOCSOut is turned on.
DOCSCustLegal	not present	not present	1 if your firm is using the DOCS Open Legal customization, 0 or not present otherwise.	Whether to support the Client and Matter fields added to DOCS Open when using the Legal industry customization.
CreateUniqueDocNames	not present	not present	1 for DOCS Open 2.5 or later, 0 otherwise	Whether to append a date and time stamp to the docname in the DOCS Open profile.
[WDData]				
path	C:\WINDRAFT\WDDATA.EXE	V:\WINDRAFT\WDDATA.EXE	V:\WINDRAFT\WDDATA.EXE	The full path of the WinDraft Data Entry program
ScriptDir	C:\WINDRAFT\SCRIPTS	V:\WINDRAFT\SCRIPTS	V:\WINDRAFT\SCIPTS	The directory in which WDData looks for scripts.
DOCSOpen	0 or not present	0 or not present	0	Whether WDData should use DOCS Open for opening and saving of files
ODMA	0 or not present	Depends on	1	Whether WDData should

		document management software		use ODMA for opening and saving of files.

[<Practice System>] (this example uses paths that might be used in an estate planning system)

DocDir	C:\WINDRAFT\EP	V:\WINDRAFT\EP	V:\WINDRAFT\EP	The main document directory
SubDir	C:\WINDRAFT\EP\SUBDOCS	V:\WINDRAFT\EP\SUBDOCS	V:\WINDRAFT\EP\SUBDOCS	The subdocuments directory
WddDir	C:\WINDRAFT\EP\DATA	V:\WINDRAFT\EP\DATA	not present	The default directory for WDData files.
BMacro	BlastOffEstatePlanning	BlastOffEstatePlanning	BlastOffEstatePlanning	The 'blast off' macro to use, provides the dialog box asking what documents to run.
Bfile	epdata.txt	epdata.txt	epdata.txt	The file to use in passing data from WDData to WinDraft
Script	EP.INT	EP.INT	EP.INT	The script to look in for a list of questions for WDData to ask the user.
DefTemplate	wdfmt.dot	wdfmt.dot	wdfmt.dot	The default Word template to use for the final document output.
AltTemplate	wdfmtalt.dot	wdfmtalt.dot	wdfmtalt.dot	The alternate Word template to use for the final document output.
AltFormat	0	0	0	Whether to default to DefTemplate or AltTemplate.
ResetToStyles	1	1	1	Whether to reset all text in the document to the formatting indicated in the Word style of the text.
FileSection	ini section if required	ini section if required	ini section if required	The file section that the names of the main documents are stored in.
WDDatatitle	your company name			For use in custom blastoff dialog boxes. Optional setting to change appearance of WDData,
barapp Text				

[System List]

System1	EP-System="Estate Planning Documents"	EP-System="Estate Planning Documents"	EP-System="Estate Planning Documents"	The first system in the list of available practice systems.
...	...	...	...	...
System10	-Corp-System="Corporate Documents"	-Corp-System="Corporate Documents"	-Corp-System="Corporate Documents"	The tenth system in the list of available practice systems.

[New System Wizard]

IntDoc	C:\WINDRAFT\SAMPINT.DOC	C:\WINDRAFT\SAMPINT.DOC	C:\WINDRAFT\SAMPINT.DOC	The WDData script to use for systems created in the System Wizard
ModelDoc	C:\WINDRAFT\SAMPDOC.DOC	C:\WINDRAFT\SAMPDOC.DOC	C:\WINDRAFT\SAMPDOC.DOC	The WinDraft model document to use for systems created in the System Wizard.

V. Appendix: Integration with DOCS OPEN Document Management

A. Introduction

Just as DOCS OPEN is "open," so is WinDraft. It is possible to set up integration so that WinDraft integrates with DOCS Open in ways that can be tailored by your firm.

There are three kinds of integration:

1. Basic Compatibility

To start with, WinDraft is *compatible* with DOCS OPEN. This means that even when you turn off other integration features, DOCS OPEN doesn't get in the way of WinDraft and WinDraft doesn't get in the way of DOCS OPEN. Each performs normally, just as you would expect.

2. Saving and Opening Data Files

Without DOCS OPEN integration, the WinDraft data entry module, WDData, displays normal DOS directories and standard Windows File Open and File Save dialogues. (WDData also supports any ODMA compliant document management system.)

With DOCS OPEN integration of the data entry module activated, WDData replaces the normal File Open and File Save functions with those of DOCS OPEN, allowing client data files to be managed by DOCS OPEN -- optionally with predefined searches.

3. Automatic Document Profile Creation

WinDraft can also automatically create and fill out profiles for each document it creates, which can be a major time saver if a lawyer creates a batch of a number of documents at a time.

If this feature is not activated, or if codes in the WinDraft Data File don't match those in the DOCS OPEN database, the document is left unsaved on the screen.

B. The User's Perspective

1. Data Files in DOCS OPEN

When WDData is set to integrate with DOCS OPEN, data files are saved in DOCS OPEN. Just as with Microsoft Word or Excel:

- File Open — Quick retrieve and profile searches
- File Save — Profile when saving a new, unnamed file
- File Save As — Profile (with version options when appropriate)

The "application" field in the profile will say "WDDATA," which is short for WinDraft Data.

The document types to be used will be those established by the MIS department, such as EPDATA for estate planning client data files and EPDATMOD for estate planning model data files.

In every other respect, DOCS OPEN behaves the same as it does with other applications, including the "bypass" functions.

2. Automatic Profile Creation

WinDraft creates batches of documents at a time. When DOCS OPEN profiling of documents is turned on, WinDraft will automatically:

- Create the document
- Verify that the information in the WinDraft data file matches the codes in the DOCS OPEN database for Client, Matter, Attorney/Author, and Document Type.
- If all the codes are valid, save the document, filling out the fields in the profile as it does so.
- Display a "hit list" of the batch of documents just created, so the user can load, edit and print.

The user can (and should) set up two saved searches to make it easy to call up documents. One, called "EP WinDraft Documents," is set with the user as typist and document type=ESTPLANWD. The other, called "EP WinDraft Documents created today" is the same, except that it also has %TODAY in the Date Created field.

C. Technical Notes

The following technical notes are for use by the MIS department, consultants, and others who are setting up WinDraft and Practice Systems for use by attorneys and secretaries.

1. The WinDraft.INI file

WinDraft.INI is the file that contains the information WinDraft and WDData use as they work to control their actions and store status information. Here are some of the special settings in WinDraft.INI that control the DOCS Open interface:

a) [WDDATA]

In the [WDData] section of the WinDraft.INI file, there should be a command that reads:

`ODMA=1`

If this line says ODMA=0, there won't be any integration with DOCS Open when WDData opens and saves a data file. If this line is not present, then WDData will use DOCS Open if present, and ignore it otherwise.

b) [WinDraft]

In the [WinDraft] section of WinDraft.INI, there are two commands that affect DOCS OPEN integration: (1) For automatic profile creation, the following line must appear in the ini file:

`SaveDMS=DOCSOpen`

and (2) WinDraft's feature to automatically know who created the document will be determined in accordance with the setting in the line UserID as follows:

- If there is a line that says

`UserID=*ENV(UID)`

the user ID will be read from the DOS environment variable "UID." You can also use a different variable by replacing UID with your choice of environment variables. You can also use a firm uses a different variable, simply insert variable name in the parentheses like the "UID" above.

- If there is a line that says

`UserID=*DOCS`

the user ID will be automatically read from DOCS Open. This line is probably the preferred solution for most firms that are running DOCS Open.

- If there is a line in the format “UserID=SmithJA,” WinDraft will use the login name from the .INI file.

c) [DOCSOpen]

This ini file section contains settings which allow finer control over DOCS Open. Your firm probably needs to customize it. For information on how to do this, please read the INI file reference appendix.

In particular, you may need a line such as
Library=LIB01

to specify the DOCS OPEN library. If this line is not present, or is set as

Library=*DOCS

then WinDraft attempts to determine this automatically from DOCS OPEN.

In addition, if your firm is running the Legal industry DOCS OPEN customizations, you will need a

DOCSCustLegal=1

line in this section.

d) [Internal]

Internal variables are set in WinDraft.INI to pass parameters from WDData to WinDraft.DOT and WinDraft.WLL. They are saved by WinDraft under the [Internal] section, and the firm need not be concerned with setting them. However, if there are problems, during troubleshooting it may be helpful to look at WinDraft.INI to see why things aren’t working as you expect. The settings of the [Internal] variables are described in detail below.

2. WDData

To set up WDData to let DOCS OPEN manage data files:

- Change WinDraft.INI so that the line ODMA=0 is changed to ODMA=1 in the [WDData] section.

- Use DOCS OPEN library management to:

a. Add the document types you want. We suggest EPDATA and EPDATMOD for estate planning data files and estate planning model data files.

b. From inside DOCS OPEN, select the Options|System Utilities|Library Maintenance menu item. Select Application Maintenance. In the presented dialog box, create a new application, called WDData. Select Filing Scheme to be Enhanced, and enter the default extension as WDD. Under Launch Methods, create a launch method to WDData.exe, and set the path properly. Set the OS Environment to Windows and the Integration type to ODMA. Once you have set the integration type, click OK, and then click DDE Settings. The DDE Settings should be as follows:

Application Name
WDDATA
Topic Name
System
Command
[Open("%1")]
Command (not running)
[Open("%1")]

Leave the print settings blank. WinDraft does not yet support printing directly from the checklist program.

After this, Click on OK, and then Close to dismiss the WDData launch method settings dialog, and then cancel to dismiss the launch methods list dialog. Next, press the Options button, and check the Shows on Desktop and Valid on Profile checkboxes.

Finally, restart DOCS OPEN and test WinDraft data entry.

It will bypass DOCS, like Word, using Shift File Open, Shift File Save, and Shift File Save As. (We suggest that typical users not be told about this).

3. Automatic Creation of Profiles

a) PROFILE Command in WinDraft Model Documents

The WinDraft language contains a command called "PROFILE." Its syntax works like this:

```
[PROFILE FieldName = "Expression"]
```

For example:

[PROFILE DocName = “Last Will of ” + GrantorName]

There should be a PROFILE command in each document for each field that needs to be filled out on the profile and is not handled by a default setting.

For example, the profile commands for a revocable trust might look like this:

[;; DOCS OPEN Profiling commands.
This creates fields WD... in Windraft.ini internal section
WinDraft automatically adds WD... to beginning of each variable in Windraft.ini]
[PROFILE DocName = GrantorName + “Revocable Trust”]
[PROFILE ClientNo = EPClientNo]
[PROFILE MatterNo = EPMatterNo]
[PROFILE Author = AttorneyLogin]
[PROFILE Abstract = “Revocable Trust of ” + HusbandName + “, for benefit of” + WifeName + “. Includes Fraction of Residue formula & GST”]
[PROFILE Grantor = GrantorName]
[PROFILE DocType = “ESTPLANWD”]

Note, that in PROFILE command you must omit two leading WD characters of the variable name. For example, the command:

[PROFILE DocName = “Husband’s Revocable Trust”]

assigns the **“Husband’s Revocable Trust”** string value to the **WDDocName** variable in the **internal** section of the **WinDraft.INI** file (see below).

It is important that MIS and the attorneys agree on document type codes and that such codes be set up in DOCS OPEN.

b) INI File Variables

As mentioned above, in order for DOCS OPEN output to work, the command [DOCSOpen=1] must be set in the [WinDraft] portion of WinDraft.INI, and the user’s login should be set up correctly (name or environment variable name code.)

In addition, WinDraft will output the fields from the PROFILE command to the [internal] section of the .INI file, preceded by the letters "WD." That is, the command 'PROFILE DocType = "ESTPLANWD"' will output a line to WinDraft.INI that says "WDDocType=ESTPLANWD."

The six variables the PROFILE macro of WinDraft.DOT currently reads are:

- current WinDraft Document Name from **WDDocName**
- DOCS OPEN Client Number from **WDClientNo**
- DOCS OPEN Matter Number from **WDMatterNo**
- DOCS OPEN Author Name from **WDAuthor**
- WinDraft Grantor Name from **WDGrantor**
- DOCS OPEN Document Type from **WDDocType**
- current WinDraft Document Group Tag from **WDAbstract**

c) How WinDraft Tracks a Batch of New Documents

WinDraft "merges" a batch of Documents at a time. From the "Blast-Off" button, the user sees a dialog box that allows the user to select from one to a large number of documents to merge at a time.

If the firm wants to modify DOCS OPEN to add an "internal" column (field) to the profile table in the database, WinDraft can use that field to store and search information about a batch. If the firm is using the standard DOCS OPEN database structure, WinDraft inserts a time and date stamp within {braces} in the abstract field, then performs a QBE search for those documents containing the time and date stamp and with App_id=WDData and Typist = UserID (usually from the environment variable, as described above).

D. Troubleshooting

1. Working with WDData Data Files

- DOCS OPEN integration isn't working. That is, when you use File Save or File Open, you see the DOS directory instead of DOCS OPEN.

Solution: Check WinDraft.INI file. [WDDATA] section must have a line that says ODMA=1]

2. Output to DOCS OPEN

- WinDraft isn't creating DOCS OPEN profiles at all.

Solution. Check WinDraft.INI file. [WinDraft] section must have a line that says SaveDMS=DOCSOpen.

- In attempting to create the profiles, an error message says "Can't open library LLDOCS01" or something similar.
  Solution: The library is not set right in the ini file. Adjust the library setting under the [DOCSOpen] section and try again.
- In attempting to create the profiles, an error message says that it can't validate Client ID, Matter ID, Attorney Login or Document Type.
  Solution: WinDraft will only create a profile if it validates each and every field in the profile against valid information in the DOCS OPEN database. There are three possible things to correct:
  1. The data entered into the data checklist for client, matter or attorney ID is incorrect or, for some other reason, it doesn't match the information in the DOCS OPEN database.
  2. There is a typographical or other error in the [PROFILE ...] command in the model document that is creating the profile entry.
  3. There is some problem at a technical level with WinDraft.INI. Call Eidelman Associates for help in troubleshooting. (This has never happened in testing, but it would be the next logical place to look.)

  In doing this troubleshooting, if the problem isn't immediately apparent, you should print out and/or edit WinDraft.INI and look in the [Internal] section to see what profile information the model document causing the problem has output. The information there *should be* the correct fields names described above in the section "INI File variables," with correct data values based on data that was entered and information from the [PROFILE] commands in the template, like this:

```
[Internal]
WDDocName=John Smith Revocable Trust
WDClientNo=99999
WDMatterNo=10022
WDAuthor=HANSONEF
WDGrantor=John Smith
WDDocType=ESTPLANWD
WDAbstract=John Smith Revocable Trust with GST
```

3. Problems Returning "Hit List" of Documents Created

When WinDraft creates a batch of documents and saves them in DOCS OPEN, it automatically inserts a {date and time} stamp in the abstract field, then does a QBE search for documents:

- with that {date and time} stamp in the abstract field
- created by the Login ID in the Typist field in the profile matching the login ID of the user.

Possible Problem: If the hit list is *empty*, there could be a problem concerning the user login of the user not matching the user login in the profiles it is searching for.

Solution: Check the WinDraft.INI [internal] section to see what UserID was output into the profiles. If it doesn't match the user ID of the person using the computer, then check the [WinDraft] section of the WinDraft.ini file. At many firms, the line should read "USERID=*ENV(UID)", which tells WinDraft to get the user ID from the DOS environment variable UID, which is set by a Network Administrator utility at login. (To see it, go to DOS and type "SET". There should be a variable on the screen like "UID=HANSONEF", which should be the same login used by WinDraft. There are also two alternate methods for setting it, described above.

VI. Appendix -- Troubleshooting

A. Macro Button Doesn't Work Right

If the same macro exists in Normal.DOT and WinDraft.DOT, it will run the version from Normal. Use File-Template-Organizer to make sure that there is not a duplicate named macro in Normal.DOT.

VII. Appendix -- Model Document Structure for EP Expert

EP Expert Estate Planning System

A. LONG FORM WILL

1. WillHusb.Doc or WillWife.Doc

Top document that calls (inserts text and/or programming from) the following:

a) GRGender.Doc

Sets values for variables such as HusbandWife

b) Will.Doc

Contains opening and closing text that is unique to wills

c) WillorTrust.Doc

Sets variables such as TrusteePr

d) TRNames.Doc

Sets variables such as Trustee1Name

e) Common.Doc

Contains text common to both long form will and revocable trust

(1) MtrNoGST.Doc

-Administration of Marital Trust article when MDGST is “N”o

--> QDOT.Doc - QDOT provisions inserted if MaritalTrustType is “QDOT”

(2) MtrGST.Doc

-Contains text for these articles: Exempt Marital Trust A and (optionally) Non-Exempt Marital Trust B. Applicable when MDGST is “Y”es

--> QDOT.Doc - QDOT provisions inserted if MaritalTrustType is “QDOT”

(3) SPOABens.Doc

-Completes the phrase: Notwithstanding the preceding provisions of this paragraph, any powers of appointment granted to my spouse hereunder may be exercised only for the benefit of ... when POAUseSpecificLang = “Y”es

f) ComPOW.Doc

Contains text common to both pour over will and revocable trust

B. REVOCABLE TRUST

1. RT_Husb.Doc or RT_Wife.Doc

Top document that calls (inserts text and/or programming from) the following:

2. GRGender.Doc

Sets values for variables such as HusbandWife

3. Trust.Doc

Contains opening and closing text that is unique to wills

a) WillorTrust.Doc

Sets variables such as TrusteePr

b) TRNames.Doc

Sets variables such as Trustee1Name

C. IRREVOCABLE TRUST

1. IRTrust.Doc

Top document for irrevocable trust.

a) Common.Doc

Contains text common to both long form will and revocable/irrevocable trust

EP Expert™

User Guide

Expert Document

Drafting Software

For

Estate Planners

from

Eidelman Associates

Microsoft®

SOLUTION PROVIDER

TABLE OF CONTENTS

1 INTRODUCTION 1-1

1.1 WHAT IS EP EXPERT? 1-1
1.2 A NOTE ABOUT WINDRAFT™ 1-1
1.3 ABOUT THIS MANUAL 1-2
1.4 CONVENTIONS 1-2
1.5 HELP RESOURCES 1-2

2 INSTALLING EP EXPERT 2-1

2.1 OPTION 1 - DOWNLOAD 2-1
2.2 OPTION 2 - INSTALL FROM DISKS 2-1

3 CREATING CLIENT DOCUMENTS 3-1

3.1 PROCESS OVERVIEW 3-1
3.2 CREATE CLIENT DOCUMENTS 3-2
3.3 DATA ENTRY NOTES AND TIPS 3-8
Screen Logic: Graying Logic in General 3-8
Special graying logic - Generation Skipping 3-8
Data Entry Tips 3-10
Entering Fiduciaries 3-11
Answer All Questions 3-13
3.4 CONTEXT-SENSITIVE HELP 3-13

4 WORD FORMATS AND STYLES 4-1

4.1 EP EXPERT "EDIT TOOLS" TOOLBAR 4-1
4.2 STYLES FOR FORMATTING TEXT 4-1
4.3 COMMON EDIT SITUATIONS 4-2
Typing in New Text 4-2
Pasting Text from another Document 4-2
4.4 FIELD CODES 4-2

5 USING EP EXPERT WITH DOCS OPEN 5-1

5.1 INTRODUCTION 5-1
Basic Compatibility 5-1
Saving and Opening Data Files 5-1
Automatic Document Profile Creation 5-1
5.2 THE USER'S PERSPECTIVE 5-2
Data Files in DOCS OPEN 5-2
Automatic Profile Creation 5-2

6 APPENDIX I - EP EXPERT COMPONENTS 6-1

1 Introduction

1.1 What is EP Expert?

EP Expert is a software enhancement to Microsoft Word™ that is designed to increase the productivity of estate planners by automating the process of document drafting. The unique user interface presents an outline of required facts (e.g., names of family and fiduciaries) and legal decisions (e.g., choice of marital deduction clause) needed to implement the estate plan, together with related data entry screens. Once you have entered and saved the data, EP Expert's "document drafting engine" produces, in Word format, the documents you select. You can then edit the documents in Word as you would any Word document.

The main benefit of EP Expert is the production of high quality, consistent documents with a minimum input of non-billable attorney time.

EP Expert contains all the basic documents used to implement a variety of estate plans, including:

- Simple Will
- Will with Testamentary Trust
- Pour Over Will
- Revocable Trust (with and without GST provisions)
- Living Will
- Powers of Attorney for Health and for Financial Matters
- Irrevocable Life Insurance Trust

See Appendix I for a complete list of client documents and other elements included in EP Expert.

1.2 A Note about WinDraft™

EP Expert is one application of WinDraft document authoring software, which works with Microsoft Word to provide checklists, data entry screens, and model documents for many legal practice areas in addition to estate planning, such as employee benefit plans and real estate transactions. For information on other applications of WinDraft that may be of interest to your firm, please contact Eidelman Associates at.734-769-1500 or *http://www.lawtech.com*.

1.3 About this Manual

This manual provides instructions for installing EP Expert and using it to produce estate planning documents. It assumes that you, the user, are generally familiar with Microsoft Windows and Microsoft Word. Except as noted, menu choices, tool bar buttons, check boxes, radio buttons, and other Windows and Word features function in their standard ways and this manual does not provide detailed instructions for their use.

1.4 Conventions

The icons and typefaces used throughout this manual to present different types of information are as follows:

How to…	A numbered procedure following this icon describes how to perform an EP Expert task.
Note:	Additional, "nice-to-know" information is shown with this icon.
Warning!	This icon warns you of a potential error.
Client Name	Names of data entry fields, selections, Web sites, and similar items are shown in *italics*.
Type **x**, **1999**, etc.	Type the exact letters, numbers, or characters shown in **boldface** text, or enter the variable indicated by words shown in bold (e.g., **userid**, **client name**.)

1.5 Help Resources

If you have questions that are not answered by this manual, you can contact Eidelman Associates:

E-mail: eidelman@lawtech.com
Phone (voice): 734-769-1500
Fax: 734-769-1501

2 Installing EP Expert

The instructions in this section are for installing EP Expert on a single-user PC. If you are installing EP Expert on a network, log on with a supervisor's ID and read the Appendices on Network Setup Notes and WinDraft.INI Settings.

2.1 Option 1 - Download

Use this procedure if you are downloading EP Expert software from our Web site.

1. Telephone Eidelman Associates to be assigned a username and password.
2. Browse to www.lawtech.com
3. Click on the "WinDraft Document Assembly" button.
4. Click on the "EP Expert Estate Planning System" hypertext link.
5. Click on the "EP Expert Download Page" hypertext link. This will bring up the "Enter Network Password" dialog box.
6. Enter your assigned username and password and select "OK."
7. Select the 95 or 97 version of EP Expert, depending upon which version you have of Microsoft Word.
8. Follow the screen instructions to run the download and complete the installation.

2.2 Option 2 - Install from Disks

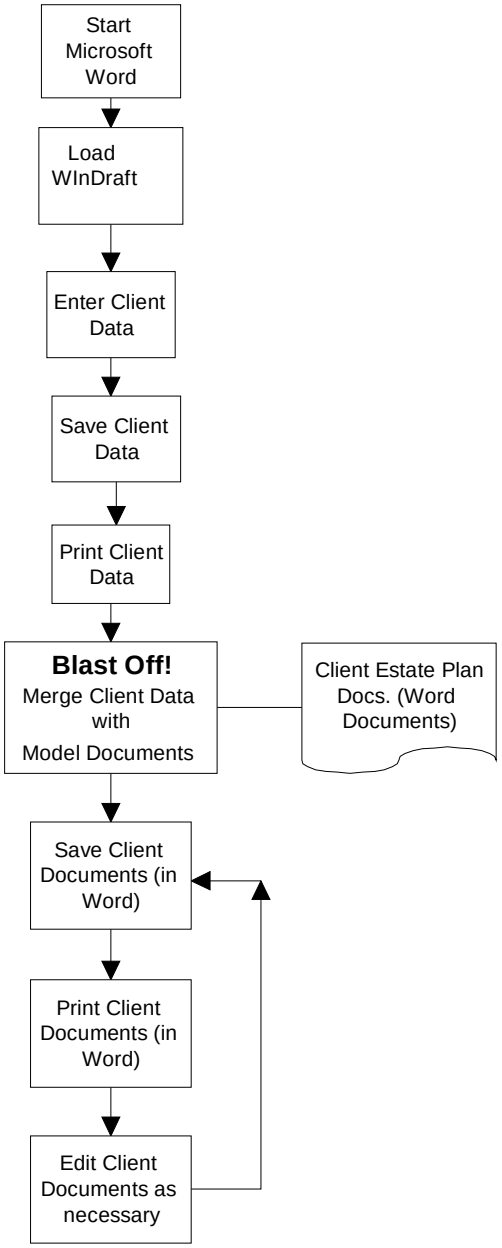
Use this procedure if you are installing EP Expert software from disks.

1. Insert the WinDraft diskette in Drive A:.
2. Close Word.
3. From Windows Program Manager or File Manager, select "File, Run."
4. Type **a:setup** and press R or click "OK." You will be prompted to indicate the drive/directory path where WinDraft will be installed. (For a single-user setup, we suggest you use the default path, i.e. "C:\Program Files\WinDraft".)

3 Creating Client Documents

3.1 Process Overview

EP Expert consists of an outline or "checklist" of client information and a set of model documents. The process for completing the checklist with information for a particular client and creating the estate planning documents for that client is shown in the following figure and described in detail in section 3.2:



3.2 Create Client Documents

This is the complete procedure for using EP Expert and client information to create all documents necessary for the client's estate plan.

Start EP Expert

1. Start Microsoft Word.

2. Click on the Load WinDraft button or press C A W to "load" the WinDraft global add-ins. The WinDraft toolbars will appear.

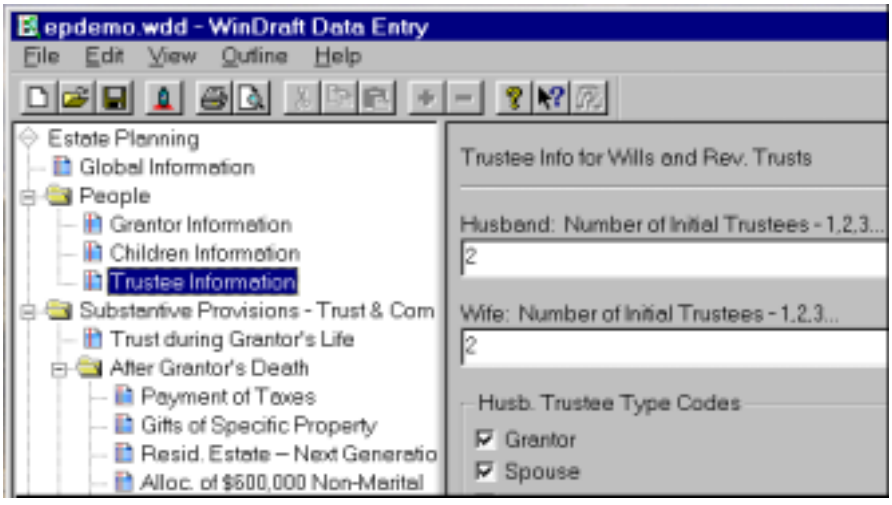
If you do not see the "Load WinDraft" button, load it by placing the cursor on any toolbar and clicking the right mouse button.

3. Click on the EP Expert "Checklist" button on the toolbar. (If it is not visible, try pressing C A W again.)

Open a Client File

1. Select "File, Open" or click on the toolbar button to open a data file. Files with a ".WDD" extension are WinDraft data files, and files with a ".WDM" extension are WinDraft model data files. You may use either.

   You will see the following screen:



 © Eidelman Associates 1999

If you are using DOCS OPEN, you will see the usual DOCS profiles, with special document types for different practice systems.

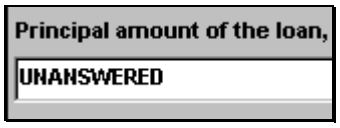
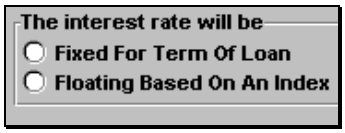
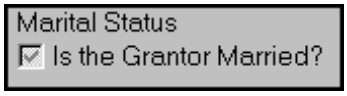
Enter Client Data

1. The data entry interface is outline-based, with topic headings on the left side of the screen and related detail data entry areas on the right. Click on the folders and pages on the left (the outline) to open the corresponding data entry areas on the right of the screen, and complete each one.

2. Use any of the following methods to move through the data entry screens:

   - Click the mouse.
   - Press the F key to move to the next field.
   - Press 4 to find the next unanswered question.
   - Press C F to move between the outline and data entry sides of the screen.

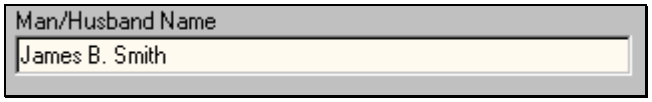
3. Unanswered questions are indicated as follows:

Text boxes	"UNANSWERED"	
Radio Buttons	None selected	
Check boxes	Shaded Gray	

If you wish to return an answer to an UNANSWERED state, use the "Unanswered" option under the Edit heading (<Ctrl-U>).

There are six kinds of data entry "controls" used to present questions: (1) text boxes, (2) check boxes and (3) radio buttons, (4) Long Text Boxes, (5) Drop Down Boxes, (6) Special Controls. They work the same way as they do in other Windows programs.

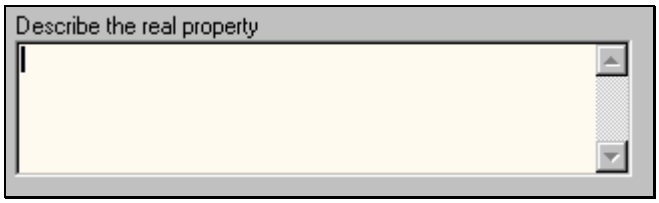
Text Boxes



Respond by typing in your answer (up to 8000 characters). If you type beyond the right side of the text box, the data scrolls like a ticker tape. Even if it is not visible, the data is all there. Use the h and e keys to move to the start or end of the text, respectively.

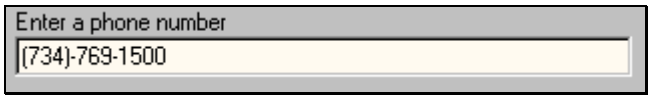
If the question is unanswered, it will say “UNANSWERED” in the checklist.

Long Text



Type your answer in, up to 8000 characters. Long Text responses will "wrap" rather than scroll off the screen as in Text Boxes. Long Text questions are also unique in their ability to accept hard returns (when you press the “Enter” or “Return” key).

Masked Text



Type your answer into a masked text box just like a normal one. Masked text boxes will accept your answer only if you enter it in the format it expects. These are often used for telephone numbers, dates, currency amounts, and other structured information.

Check Boxes

Check boxes are used to present questions calling for a Yes/No response. They can have three states:

- Checked, which means Yes

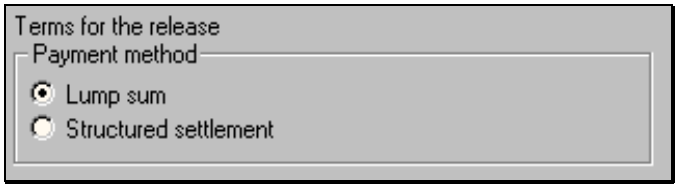
- Unchecked, which means No

 © Eidelman Associates 1999

- Hatched, meaning Unanswered

Checkboxes sometimes come in groups. You may check more than one option.

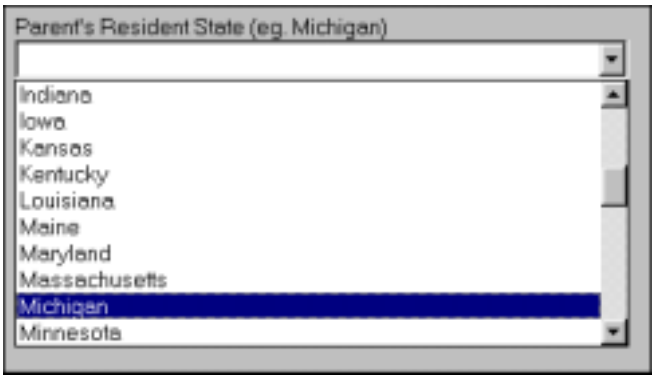
Radio Buttons



Radio buttons are used for multiple choice options. Only one option can be selected. If one choice is already selected, click on another to de-select the first choice.

If the question is unanswered, all of the buttons will be displayed in white.

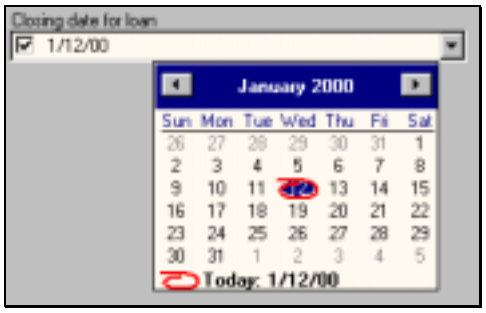
Drop-Down Box



A drop-down box is similar to radio buttons in that it offers a choice of one item from among many alternatives. However, not all choices are immediately presented. To make a selection, click the drop-down button, use the scroll bar to highlight your answer, click on the answer. If the question is unanswered, the drop down box is empty, like an unanswered long text box.

There are two types of drop-down boxes in EP Expert: One limits your choice to the items in the drop-down list; the other allows you to either make a selection from the list or type in text of your own.

Dates



A calendar control may be used for entering dates. This looks similar to a dropdown box, but it has a box at its left which is checked when the control has been answered. Clicking the arrow at the right of the textbox drops down a calendar.

Save your data

Use "File, Save As" to save the modified data file under any name you choose, followed by .WDD for client data. (We suggest using the client's last name as the filename for easy recognition of the file.)

You will have other file-saving options, depending upon whether or not you are working with document management software such as DOCS OPEN or Eidelman Associates' *Law Office 97.*

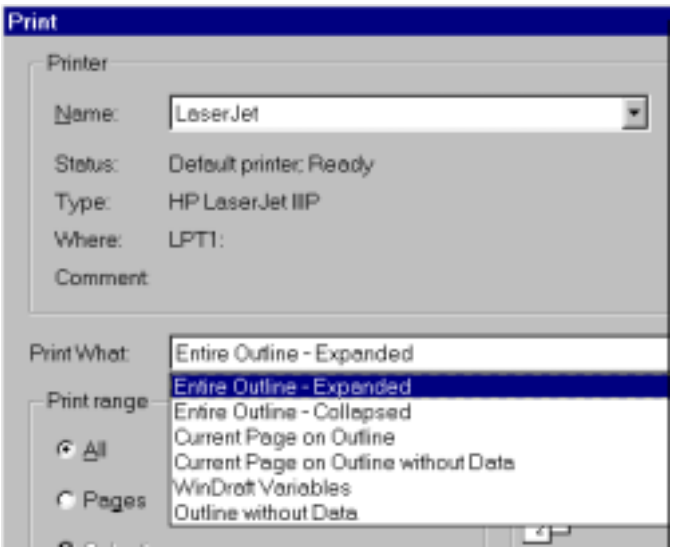
Print your data

1. Either click on the "Print" button **or** select "File, Print" from the menu.

2. To preview the print output on the screen, select "File, Print Preview."

There are six printing options:

 © Eidelman Associates 1999

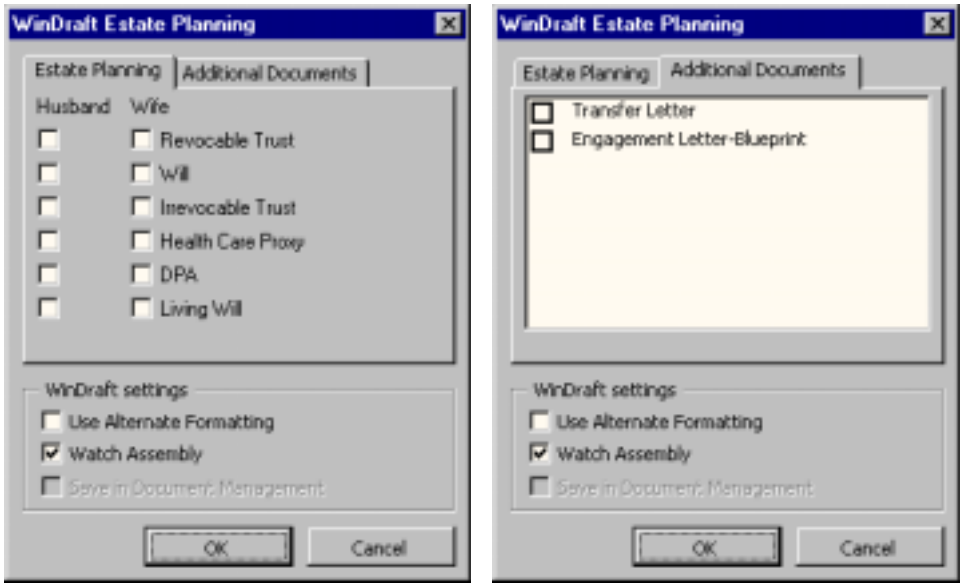


Normally, you will print the expanded outline to see your data presented in a "lawyer-friendly" way. **We recommend placing a copy of the data outline printout in the file** so that a lawyer proofreading the documents can easily verify the selected options.

To print a blank outline for use by lawyers not on the system (or for use in a client interview), select the option "Outline without Data."

BLAST OFF!

1. Click on the "Rocket" icon to switch to Microsoft Word and access the "Blast Off" dialog box:

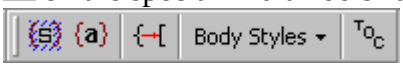


2. Select the documents you want the system to create, and click "OK."

Although it will take slightly longer, we recommend that you select "Watch Assembly" so that you can observe the document as it is assembled on the screen.

3. The system will merge the model documents and the client data you entered to create the selected documents. The system will display a status box to show progress during the merge process, with the message "Document Assembly Completed!" when done.

4. If you wish to create a Table of Contents for your documents, click the "TOC" button on the special "Edit Tools" toolbar.



5. Save the documents as you would any Word document.

3.3 Data Entry Notes and Tips

This section presents general information on data entry screen logic and tips that you will find useful as you enter client data.

Screen Logic: Graying Logic in General

The data entry screens are programmed to "gray out" certain choices. For example, if there is a QTIP marital, the spouse cannot have a right to withdraw the principal, so that option is grayed out.

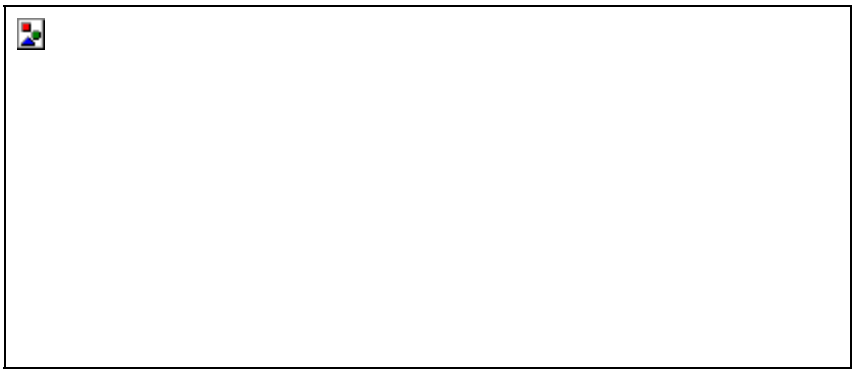
If a question is grayed out on the screen, it is disabled, and it will be skipped when you navigate through the screens. This is true with any of the navigation methods.

Special graying logic - Generation Skipping

Special trust options to avoid the generation skipping transfer tax are presented on the "full options" data entry screens.

To "turn on" the GST options, answer **Yes** to the "Generation Skipping Trusts?" question on the "Documents for Client" screen:

 © Eidelman Associates 1999

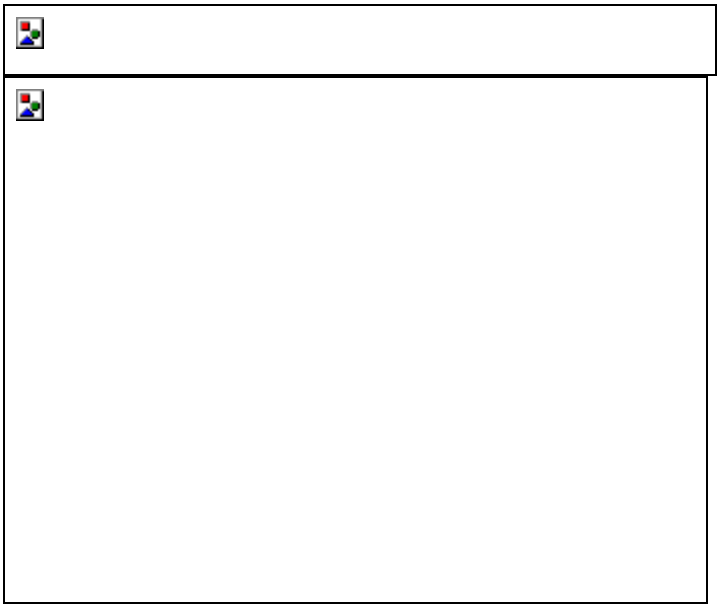


If you answer **Yes** to GST, the later questions relating to the specifics will be turned on so that you can answer them.

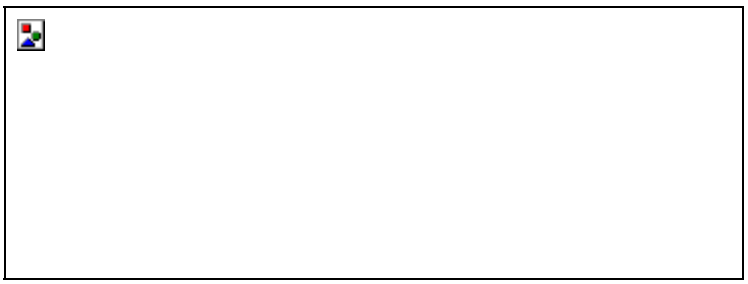
If you answer **No** to GST, the related questions will be grayed out.

The GST questions include:

- GST-exempt Marital Trust ("Reverse QTIP")



- GST options in "Trusts for Children" section.



Data Entry Tips

Select the Simplest Level Possible

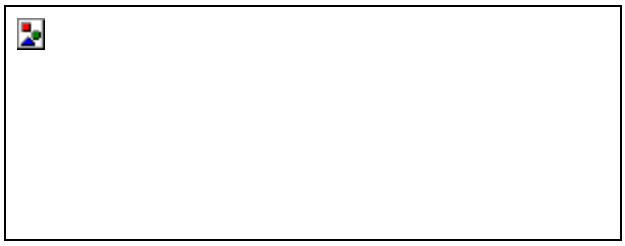
The set of model documents contains ***all*** of the possible language needed to produce sophisticated documents. However, in order to simplify data entry, we have set up alternative WinDraft Checklist templates with fewer questions.

How to Access the Correct Checklist

1. Click on the "Change Practice Systems" button.

2. Select the simplest level that fits your client.

 © Eidelman Associates 1999



You *can* use the full data entry screens to draft a simple will, but you would have to wade through dozens of irrelevant screens. It is best to work with the simplest set of screens that fit your client's needs.

Start with a data file that is already full of data

In normal operation, you will never select "File|New" in the data entry screens. Rather, you will start with a pre-answered data file, then "Save|As" using a different name. The file you open should be the one that comes closest to matching the needs of your client. It can either be a standard answer file for the firm, or it can be one created for another client whose needs are the same as your current client.

These will be well organized using DOCS Open, but for now it is up to the firm to organize and name these data files. With DOCS, the system can be set up to automatically only show documents that satisfy a certain search request when you run File|Open. In this way, File|Open will only list the sub-set of data files of interest to you.

Entering Fiduciaries

Free-text format for fiduciaries:

When entering fiduciaries in a list, enter complete information, including, if you want, any descriptive information, such as the person's relationship and/or city. If you want the name in all caps, you must format it that way here.

- All Initial fiduciaries are all listed in the first box, and successors begin with box number 2. If there are co-fiduciaries, enter them in the same field.
- If there are co-fiducaries, be sure to put "or the survivor of them," or the equivalent, after their names.
- The system is not programmed for special provisions (e.g., if one co- doesn't serve; appoint a bank as co-, etc. You must edit the draft documents to insert such provisions.

- Do not put any commas or other punctuation at the end of an entry.
- In the document, fiduciaries will be presented in a numbered list or separated by semicolons in a paragraph, depending upon the style your firm has chosen to use.
- Note that names in the free-form text fields will NOT be automatically capitalized. As you type them in, you must format your entries EXACTLY as you want them to appear in the documents.

Entering Repeating Data

The program keeps count of how many fiduciaries you have entered; you do not need to indicate the total number. Simply enter as many as you want, and leave the next one UNANSWERED.

Examples:

My wife, Sally
My brother, David A. Smith
My attorney and friend, PERRY MASON
John A. Johnson, now residing in St. Louis, Missouri
My brothers, BILL SMITH and ANDY SMITH, as co-Personal Representatives, or the survivor of them

Using "Same as Husband" option

If you option to use the same fiduciaries in the wife's documents as you entered in the husband's, do not enter the relationship information, as "brother" will not automatically be converted to "brother-in-law."

Note that if you answer "Yes" to "same as husband?" and the Husband had named the Wife, then the system **will** automatically do the mirror image, with the wife appointing the husband.

 © Eidelman Associates 1999

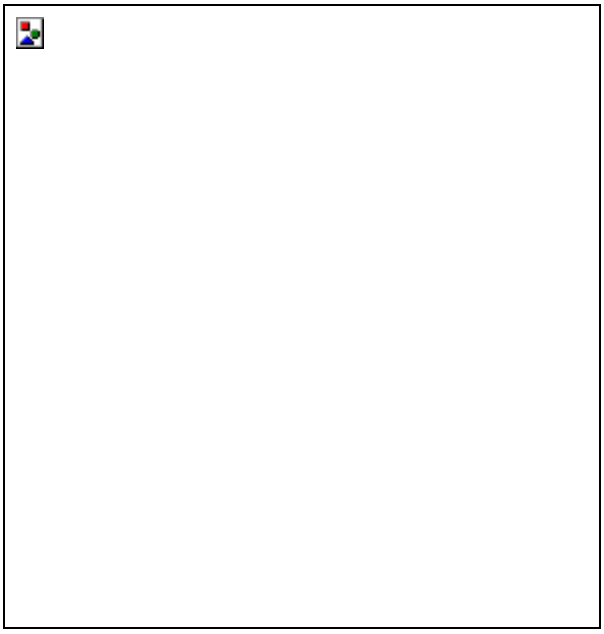
Answer All Questions

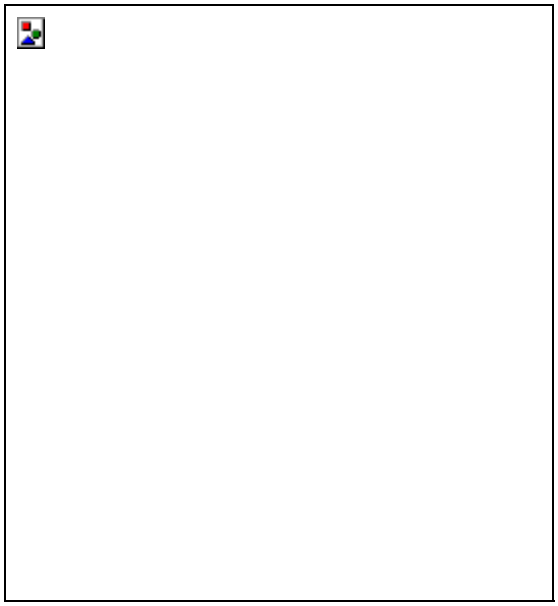
Be sure to answer all questions. Note that a text box containing the word "UNANSWERED" indicates a question that has not been answered; this is *not* the same as having answered the question "NO."

Unanswered questions will cause errors when you Blast Off!

3.4 Context-sensitive Help

An asterisk next to an outline item or data entry field indicates that legal help is available on that subject. Click on the "Legal Help" button to bring up the related help screen.





- If the cursor is located on the outline on the **left** side of the screen, clicking on "legal help" will bring up the help screen related to the "page" of information **related to that point in the outline.**

If the cursor is on a field on the **right** side of the screen (in a text box, check box or set of buttons), clicking on legal help will bring up the help screen **related to that question**, if there is help for the question. If not, this action will bring up the help screen related to the page of information for the outline heading (left side).

The Help screen will pop up as either:

- a window in the middle of the screen that contains plain text; or

- a Web browser (Netscape or Internet Explorer) with legal help in HTML (Web browser) format. (This is because the help file is an HTML file, with links to various websites. If you are not logged onto the Web and the help you request includes one of these links, you will get an error message.)

 © Eidelman Associates 1999

4 Word Formats and Styles

Every Word document is attached to .DOT file -- a template -- consisting of styles that control the formatting of the document. EP Expert uses this concept of styles and templates to assure consistent and correct formatting of all client documents. This formatting occurs automatically as you enter client data and "blast off" to create client documents. The only time you need to be aware of how styles apply within EP Expert is when you edit a client document. This section describes the special format functions (e.g., specific styles and unique toolbars) that are part of EP Expert and explains how they affect some of the most common editing situations.

For more information on styles and how to use them, see your Microsoft Word documentation.

4.1 EP Expert "Edit Tools" Toolbar

EP Expert makes extensive use of Microsoft Word features such as bookmarks, field codes, automatic paragraph numbering, table of contents, cross referencing, outline headings, and text formatting. All of these functions are available through the standard Word dropdown menus; the special toolbar simply provides "shortcut" buttons.

If the special Edit Tools toolbar is not visible, go to WinDraft Checklist and choose the View | Options menu. In the resulting dialog box, check "WinDraft Editor Toolbar" and click "OK." You may need to restart Word to display the toolbar.

4.2 Styles for Formatting Text

Text within the client document is formatted according to "styles." These styles control the appearance, but not the content, of the text. Styles are also used by such functions as paragraph numbering, table of contents and index generation, and cross-referencing. Therefore, if you want to change the font, font size, or formatting of the text, you **must** do it by applying a style or changing a style.

Because all text takes on the style of the paragraph in which it resides, it is a good idea to work with the "Show/Hide ¶" feature turned *on*.

4.3 Common Edit Situations

Some of the most common editing situations in which you need to be aware of styles are described in this section.

Typing in New Text

Remember that new text will take on the style of the paragraph into which it is inserted. If you place your cursor at the correct insertion point before you begin typing, the new text should be formatted correctly. However, if you need to type in an entire new paragraph, or a new heading, or some other element, you will need to apply the appropriate style to it, as follows:

1. Type the text.

2. Block the new text, along with its ending paragraph mark (¶).

3. Click on the arrow in the "Styles" box on the Word toolbar.

4. Move through the list of styles until the one you want is highlighted.

5. Click again to apply that style to the new text.

Pasting Text from another Document

This is the most common editing situation in which you must be aware of styles. When you cut and paste text from another document, the style of the previous document will come with it *if* you included the ending paragraph mark. If not, it will take on the style of the paragraph into which you paste it. To be sure that the pasted text is formatted with the correct style, select it and apply the style as described above for inserting new text. Be sure to select the ending paragraph mark.

Word includes nine heading styles and nine body styles. (Body styles apply to the paragraphs following a heading, until another heading style is applied.) EP Expert uses Heading1 - Heading4 and Body1 - Body4.

4.4 Field Codes

EP Expert uses Word field codes to change certain terminology in a document (e.g., husband/wife, he/she, trust/will). If you insert text into a document (by either typing it in or cutting and pasting), or delete text, you must be aware of fields that may be included in that text.

 © Eidelman Associates 1999

- Click on the show field shading ( ) button to turn on and off the shading of Word fields to make them easily identifiable.

- Press 9 to "recalculate" fields after making any changes to a document.

See your Word documentation for more information on field codes.

5 Using EP Expert with DOCS OPEN

5.1 Introduction

EP Expert and other WinDraft practice systems can be used with any ODMA compliant document management system, such as DOCS OPEN. This section describes the recommended ways of using EP Expert and DOCS OPEN together.

There are three kinds of integration:

Basic Compatibility

EP Expert is *compatible* with DOCS OPEN. This means that even when you turn off other integration features, DOCS OPEN doesn't get in the way of EP Expert and EP Expert doesn't get in the way of DOCS OPEN. Each performs just as it would if the other were not running.

Saving and Opening Data Files

Without DOCS OPEN integration, the WinDraft data entry module, WDData, displays normal DOS directories and standard Windows File Open and File Save dialogues. (WDData also supports any ODMA compliant document management system.)

With DOCS OPEN integration of the data entry module activated, WDdata replaces the normal File Open and File Save functions with those of DOCS OPEN, allowing client data files to be managed by DOCS OPEN -- optionally with predefined searches.

Automatic Document Profile Creation

WinDraft can also automatically create and fill out profiles for each document it creates, which can be a major time saver if a lawyer creates a batch of a number of documents at a time.

If this feature is not activated, or if codes in the WinDraft Data File don't match those in the DOCS OPEN database, the document is left unsaved on the screen.

5.2 The User's Perspective

Data Files in DOCS OPEN

When WDdata is set to integrate with DOCS OPEN, data files are saved in DOCS OPEN. Just as with Microsoft Word or Excel:

- File Open Quick retrieve and profile searches
- File Save Profile when saving a new, unnamed file
- File Save As Profile (with version options when appropriate)

The "application" field in the profile will say "WDDATA," which is short for WinDraft Data.

The document types used will be those established by your firm's MIS department, such as EPDATA for estate planning client data files and EPDATMOD for estate planning model data files.

In every other respect, DOCS OPEN behaves the same as it does with other applications, including the "bypass" functions.

Automatic Profile Creation

WinDraft creates batches of documents at a time. When DOCS OPEN profiling of documents is turned on, WinDraft will automatically:

- Create the document
- Verify that the information in the WinDraft data file matches the codes in the DOCS OPEN database for Client, Matter, Attorney/Author, and Document Type.
- If all the codes are valid, save the document, filling out the fields in the profile as it does so.
- Display a "hit list" of the batch of documents just created, so the user can load, edit and print.

The user can (and should) set up two saved searches to make it easy to call up documents. One, called "EP WinDraft Documents," is set with the user as typist and document type=ESTPLANWD. The other, called "EP WinDraft Documents created today" is the same, except that it also has %TODAY in the Date Created field.

 © Eidelman Associates 1999

6 Appendix I - EP Expert Components

FILE INFORMATION

- Office Identification
- Client File Identification
  - File Name
  - Client Number
  - File Number
- Lawyer Identification

CLIENT INFORMATION

- Married Couple
  - Man
  - Woman
- **Man/Woman**
  - Full Name
  - Informal Name
  - Address
  - Citizenship
  - Married
  - Children
- **Notary State/County Blank**
- **Children**
  - Number
  - Full name
  - Informal Name
- **Appoint Guardians**
  - Names

Appoint Conservators

- Names

CLIENT DOCUMENTS

- **Man/Woman**
  - Pour Over Will
  - Revocable Trust
    - GST Provisions
  - Living Will
  - Financial Power of Attorney
  - Health Care Power of Attorney
  - Irrevocable Trust
  - Will with Testamentary Trust
  - Simple Will

FIDUCIARIES

- **Personal Representatives**
  - Initial
    - Number of Individuals
    - Corporate
      - Names
  - Successors
    - Number of Individuals
    - Corporate
    - Named by Testator
    - Named by Exiting PRs
      - Names
- **Trustees - Rev Trust**
  - Same as PRs
  - Initial
    - Number of Individuals
    - Corporate
      - Names
  - Successors
    - Number of individuals
    - Corporate
    - Named by Grantor
    - Named by Existing Trustees
      - Names
- **Agents**
  - Financial PO
    - Same as PRs
      - Names
  - Health POA
    - Same as PRs
      - Names
- **Trustees - Irrevocable Trust**
  - Number of individuals
  - Corporate
    - Names

DISPOSITIVE PROVISIONS

- **Trust During Testator/Grantor's Life**
  - Allowable Distributions
    - Family
    - Children
- **After Testator/Grantor's Death**
  - Payment of Taxes
  - Specific Property
    - Residence
    - Tangible Personal Property
      - Written Statement
      - Specific Gifts
  - Credit Shelter/Marital split
    - Marital Deduction Formulas
      - Various Pecuniary Formulas
      - Fraction of the Residue
  - Non-Marital Share
    - Trust with Life Estate for Spouse
    - To Issue - In Trust or Outright
  - Marital Share
    - Trust
    - Outright
    - Disclaimer
      - GST and Non-GST Trusts (QTIP or POA
    - Distribution of Principal
      - To Whom
      - Standard
    - Special Powers of Appointment
  - Next Generation
    - Outright
    - Separate Trusts or POT
      - Ages for Distribution

ADMINISTRATIVE PROVISIONS

- Fiduciary Powers
- General Provisions
- Miscellaneous Provisions
- Presumption of Survival
- Powers of Appointment

OPTIONS THAT AFFECT WILL

- Tax and Administrative Options
- Personal Representatives
- Residuary Estate

HEALTH CARE POWER OF ATTORNEY

- Agents

FINANCIAL POWER OF ATTORNEY

- Agents

IRREVOCABLE TRUST

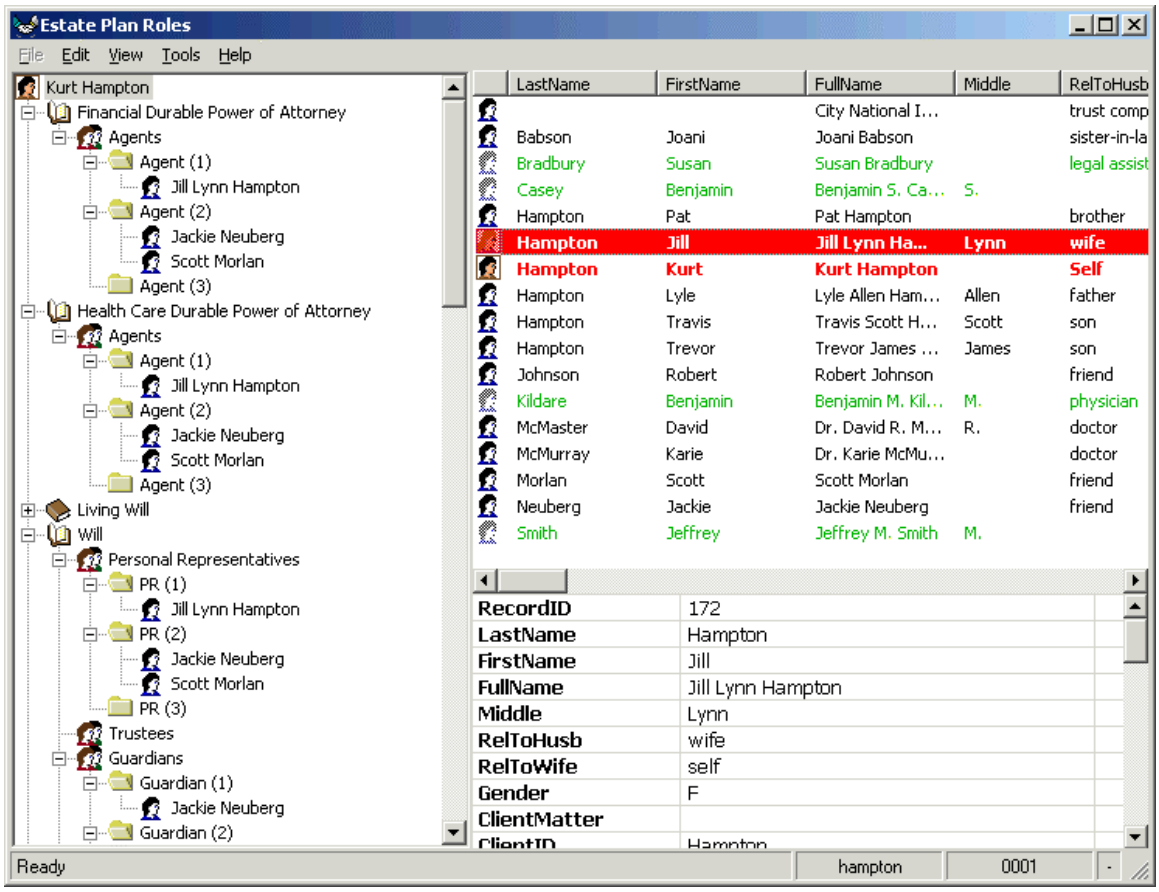
- 1st or 2nd Death
  - Contingent Marital
  - Single or Joint Grantors
- Who has Right of Withdrawal
- Trustee
  - Successors

 © Eidelman Associates 1999

Eidelman Associates

EP People™
Fiduciary Chooser™

User Manual



Eidelman Associates

November 12, 2001

© Eidelman Associates 2001

EP Expert Fiduciary Chooser
Table of Contents

I. INTRODUCTION 1

A. Documentation. 1

B. This is a Work in Progress 1

C. Why EPPeople? 2

II. THE SOLUTION 2

A. EP Fiduciary Chooser 2
1. Screen – What it looks like 2
2. Benefits to Users 3
3. Benefits to System Authors 4
4. The Technology 4

B. Entering the People 5
1. Architecture and Benefits: 5

III. WORKING WITH THE EPPEOPLE.EXE PROGRAM 5

A. EPPeople Automatically Linked to the WinDraft Checklist 5

B. EPPeople as a stand-alone Program 6

IV. WORKING WITH DATA 6

A. Clients and Matters 6

B. Working with People 7
1. Table View Screen: Enter and view data in table view: 8
2. Record View Screen: A form for one record at a time 8

C. Working with the Roles Tree 9
1. Select a Client-Matter 9
2. The Roles Tree 10
3. Drag and Drop 11
4. Copying Branches of the Tree 12
   a) Copy any Node to any Like Node 12
   b) Mirror Image Copying 13
5. Working with the Tree Display 14

D. The People List 15

E. Saving Your Data 15

V. BLAST-OFF TO OUTPUT YOUR DATA TO WINDRAFT 16

A. How to Blastoff 16

B. What Happens 16

C. Blastoff Options 16

VI. PRINTING THE EPPEOPLE 17

VII. EPPEOPLE TECHNICAL NOTES 22

A. Authoring – Programming the Documents 22
1. Getting the Data into the Documents using GETDATA 22
2. "Scope" of the Variables 23
3. Using Personal Pronoun Functions 24
   a) How the functions work 24
   b) Using the functions with EPPeople 24
4. Using Word fields 26
5. Using the Fiduciary Variables in Context 27
   a) Without FOR Loops 27
   b) With FOR Loops 27
   c) Combining The Two Approaches 27

B. Authoring – Programming the Checklist 28
1. Use VBScript or JavaScript programming in a .WDS file to have variables and Events in WinDraft's checklist interact dynamically with EPPeople.EXE. 28
2. Events that interact with EPPeople can include Windraft system events (Blastoff, Save, Open), or can be programmed to react when the user exits from a field or clicks an Author-defined button. 28
3. You can also interact with EPPeople's Database or XML files from other parts of the system, such as blastoff programs in Word VBA, etc. 28

C. Authoring – Editing Templates 28
1. Understanding EPPeople Templates and Data Files 28
   a) First, open the data file or template file. 28
   b) Second, right mouse click on the appropriate level and make the changes you want. 28
   c) Finally, save the data file or template. 28
2. How to create a new role or document 28
3. How to remove a role, document or a grantor 30
4. How to change the properties of a node 31
5. How to reorganize the tree structure 31
6. Summary 31

D. "Under The Hood" -- Working with EPPeople Role Files 32

1. How the Files Fit Together 32
a) About XML 32
b) General.EPT 33
c) Client .EP files: 33
d) Technical Notes on Working with the XML and MDB files 35

VIII. NEW FEATURES NOT YET DOCUMENTED 36

IX. WISH LIST 37

X. APPENDIX – SETUP AND TROUBLESHOOTING 38

A. Setup and Architecture 38
1. 3 Parts of Setup (These may be combined into one or more setup routines. 38
a) Set up DLL's and registry settings to make the Windows installation compatible with all of tools being used. (All Microsoft. They are latest versions of ADO, XML, etc.) 38
b) Set up Application files you need to run the EPPeople part of the application. They go on Drive C: if you are a typical local user, and on a network "share" (a shared directory) if you are working on a network. 38
c) Programming or setting up the practice system variables and WinDraft scripting to "lock into" the EPPeople way of doing it, including stylistic and legal drafting issues. 38
2. Troubleshooting: Temp Blastoff Text Files on Temp Directory 38
a) Temp File Output Location. 38
b) Making your Temp files Visible 39

B. Note on Database Connections: 40

C. Troubleshooting EPPeople Issues – Following the data 40
1. Database Entries 40
2. The Roles Tree 40
3. "Blast-off" Temp Text Files 41
4. Variables Merged into the Documents 41

D. Setup Procedures on Windows 95 and NT 42

E. MDAC Drivers 46

XI. AUTHORING -- OTHER MIRROR-IMAGE FUNCTIONS 46

XII. APPENDIX – MORE AUTHORING EXAMPLES 47

A. Will 47

XIII. AUTHORING – THE EPPEOPLE OUTPUT 48

A. Getdata Commands 48

B. Examples **49**
1. The User's View: 49
2. The Blastoff File temporary Data 49
a) In WDOutputAll.txt 49

C. In Husb.FDPOA.txt **53**

Eidelman Associates

EP People
Fiduciary Chooser for EP Expert™

I. Introduction

A. Documentation.

When you first use the program, you should review the "*Quick Start To Using EPPeople*," which contains a step-by-step list of what to do to create the data and assign the roles for an estate plan. It is not a substitute for this documentation, however.

Also, there is a list of undocumented features or various notes and screen shots to be incorporated into the manual at the end of this documentation.

B. This is a Work in Progress

Our EPPeople module of EP Expert and this documentation are works in progress. As technology changes, and as more law firms get experience with it and request additional features, this program will continue to be enhanced in a number of ways. These enhancements may include the estate planning functionality, the way it integrates with the WinDraft environment, its user interface and programming interface, and the way the setup/installation works. Consequently, do not be alarmed if the version you are working does look or work exactly as described in this document.

Please visit www.lawtech.com/windraftupdate for further information. And please send us an email, or give a call, as you find issues or have suggestions, either as to the program itself or the documentation.

We believe that this manual and the combination of the examples we have set up for your practice system will be sufficient to allow both users and WinDraft System Authors to work effectively with the program. If there are items that are new enough that they haven't been documented, we apologize, and thank you for working with us as we add new features. We believe that it is better to use our latest (tested) work than programming with an older version, even if the documentation isn't yet complete.

© Eidelman Associates 2001

C. Why EPPeople?

EPPeople solves a difficult document programming problem. In the past, it has been extremely complex and tedious to program fiduciaries for estate planning documents, especially when "mirror image" documents for Husband and Wife are required.

In programming estate planning documents, half of the battle is dealing with people and their roles in the different documents. Simply stated, it is complicated to a degree that far exceeds the amount of effort it takes to solve the problem one project at a time.

Database designers speak of "many-to-many" relationships, and here we have many-to-many-to-many-to-many-to-many. There are husbands and wives, each of whom have many documents, each of which have many fiduciary roles, each of which can have an array of people associated with them (many successors, and at each level of succession, there can be one or more co-fiduciaries), and each person may have a different relationship to the husband and wife.

That complex data structure is compounded by the need to be able to insert this information into documents with great flexibility in the way the data is combined and formatted. The way the information is merged into the documents may vary from state to state, firm to firm, and attorney-to-attorney, depending on the law and the drafting style of the attorneys.

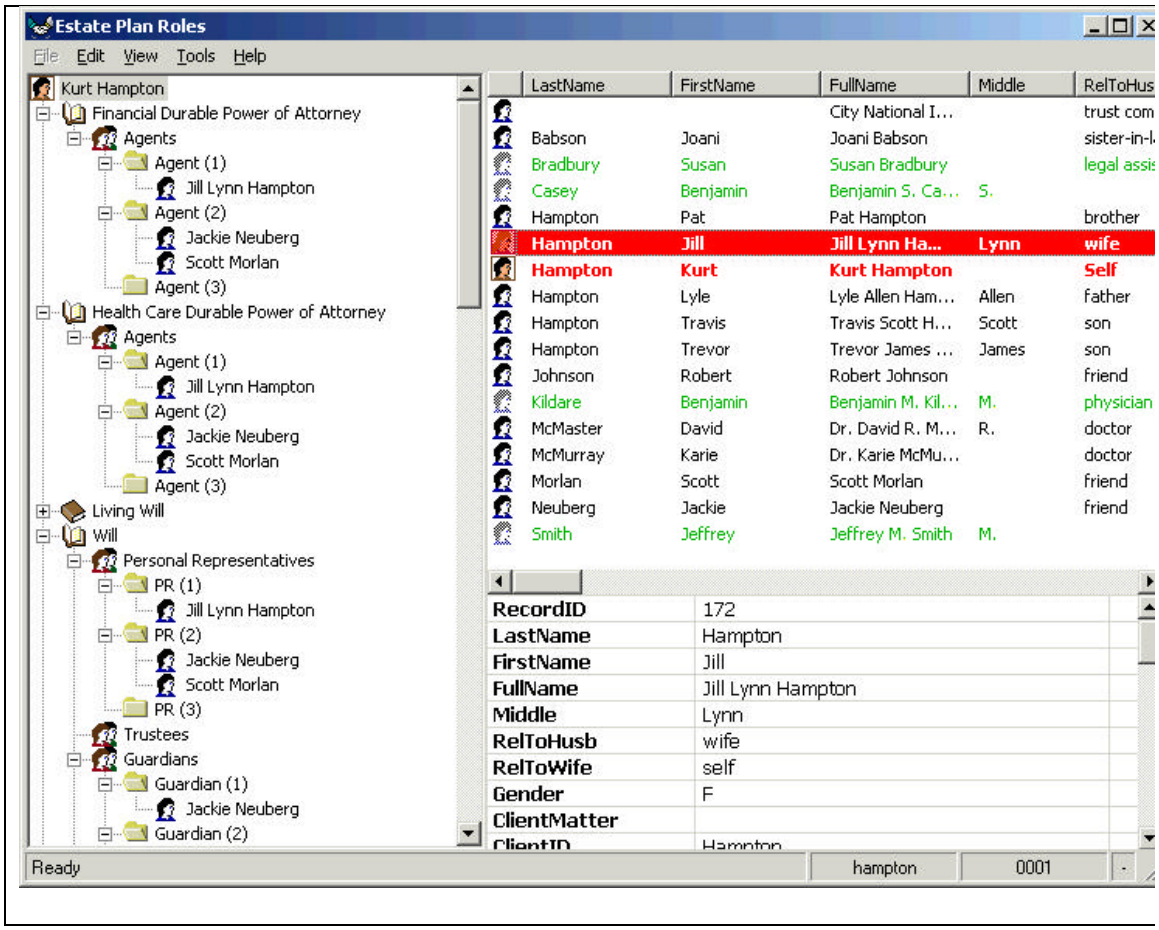
II. The Solution

EP Expert now includes a flexible, user-friendly approach to managing the information about people and their roles in the documents; it is called EPPeople. It combines an Windows Explorer-like "tree control," drag-and-drop interface, and advanced programming approaches using XML[1] and COM[2].

A. EP Fiduciary Chooser

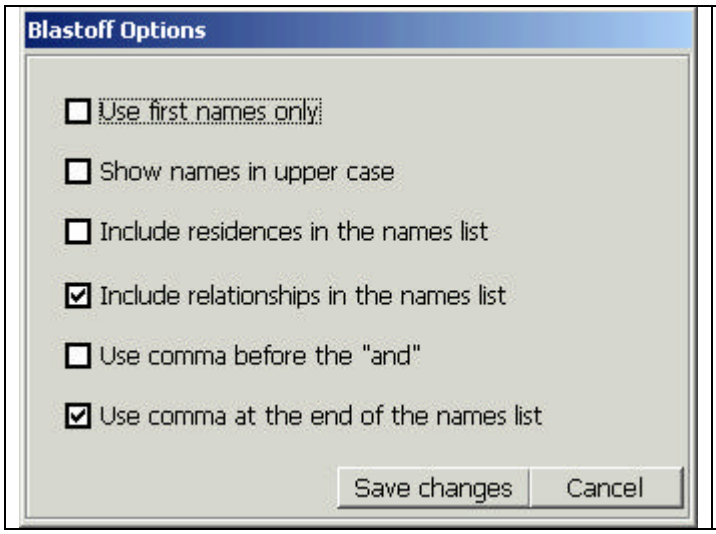
1. Screen – What it looks like

[1] XML – Extensible Markup Language – See www.xml.org, www.msdn.microsoft.com/xml, and "Why I like XML," http://www.xmlmag.com/upload/free/features/xml/2000/04fal00/dw0004/dw0004.asp.
[2] COM – Microsoft's "Component Object Model" architecture. See http://www.microsoft.com/com/.



2. Benefits to Users

- User-friendly
- Powerful
- Fast
- Only enter each name once, then drag-and-drop the person to the role in the right document.
- Automatically handles punctuation and wording
- Offers great flexibility in the formats you can use to output data into a document:


Blastoff Options
Use first names only
Show names in upper case
Include residences in the names list
Include relationships in the names list
Use comma before the "and"
Use comma at the end of the names list
Save changes
Cancel

Mix and match the options!

Examples: The phrases below are variable outputs that can fill in the blank in the following sentence. "I appoint _____ to serve as initial Trustee."

Depending upon the options chosen, here are some ways EP Expert can automatically fill in the blank:

- John A. Smith
- John A. Smith, Fred Smith, and Sally F. Johnson
- my wife, Sally Johnson, my brother, Fred Smith, and my sister, Jane McMurphy
- my brother, ROBERT SMITH, residing in St. Louis, Missouri, and my sister, SALLY F. ROGERS, now residing in San Francisco, California.

3. Benefits to System Authors

- Easier and faster to program
- Consistent naming convention for fiduciary variables
- Great flexibility
- Lower cost
- Removes drudgery from dealing with fiduciaries.

4. The Technology

Uses the modern XML format to store data about the roles. This offers dozens of advantages, including flexibility, compatibility, and ease of use. The programs are written in Microsoft Visual Basic, and interact with your

choice of database. (By default, it's an Access 2000 MDB file (Jet 4.0), but it could also be Access 97, SQL Server, or others).

B. Entering the People

1. Architecture and Benefits:

The people are stored in a database table. This has several advantages:

- Easy to enter the data
- Enter each person's information just once
- Use the data for other purposes, such as marketing, conflicts of interest searching, and matter management.
- Compatible with other databases the firm uses.

The data is typically stored in Access or SQL Server tables.

III. Working with the EPPeople.exe Program

In normal use, the program will be accessed using the first method described below, as an ActiveX program called from within WinDraft's Wddata.exe checklist.

EPPeople can be accessed in two ways:

A. EPPeople Automatically Linked to the WinDraft Checklist

EPPeople.exe is an ActiveX component in the Microsoft COM architecture. (See www.microsoft.com/com.) COM stands for "Component Object Model," and is an architecture by which one application can "talk" to another application programmatically.

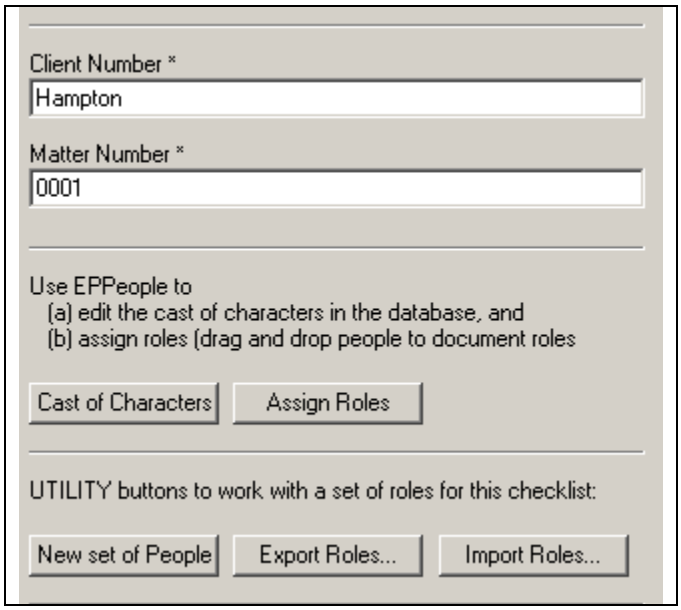
WinDraft, through its VBScript interface (WinDraft 4.0 and later), and EPPeople both support COM, and can talk to each other programmatically. That is, WinDraft events (button clicks, blastoff, etc.) can launch a VBScript subroutine that tells EPPeople what to do, and vice versa.

The WinDraft practice system checklist now has automatic events and buttons on the WinDraft checklist that activate automatic File New, File Open, and File Save functions so that the user does not need to worry about what the EPPeople file is named, where it is saved, etc. The EPPeople data is saved right within the WinDraft WDD file when it is saved (using Save As), so that neither the user nor any document management system need to worry about managing them

Import/Export Role Data

In normal operation, File|New, File|Open, File|Save, etc. are handled in the WinDraft checklist, and they will automatically create, save and close the

EPPeople Roles data files, saving the data within the WinDraft checklist. If you need to import and expert the people information from the WDD file, there are now buttons on the checklist for that:


Client Number *
Hampton
Matter Number *
0001
Use EPPeople to
(a) edit the cast of characters in the database, and
(b) assign roles (drag and drop people to document roles
Cast of Characters
Assign Roles
UTILITY buttons to work with a set of roles for this checklist:
New set of People
Export Roles...
Import Roles...

B. EPPeople as a stand-alone Program

You can also run EPPeople like any other program. That is, it can be launched from the Start|Programs menu, or from Windows Explorer. It has its own File Open, File Save/Save As functions, and to output data to WinDraft, it has its own File|Blastoff function.

To use it in this way, separately launch EPPeople, work with it as a separate program, and use the Windows Taskbar or Alt-Tab to move between WinDraft, Word and EPPeople, just as you do with other programs.

IV. Working with Data

EPPeople can work with data in a firm's own database or case management system, or even with Microsoft Outlook. If set up that way, then your firm may enter clients, matters and people using a different interface.

The examples below assume that you are adding and editing clients, matters and people through our very basic interface described below.

A. Clients and Matters

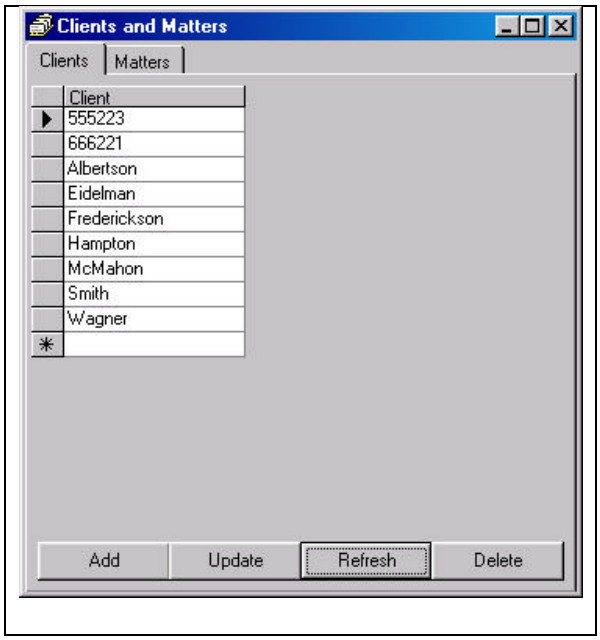
Before you can add people for a client/matter, you need to add a client and matter code.

In normal operation, you add the client and matter code in the WinDraft interface under the File Info screen. When you click the "Assign Roles" button, the system

will automatically create a new EPPeople roles file with the correct client and matter automatically inserted into the database.

Manually, you can also click on the View|Clients and Matters menu choice, and then add a client and matter by clicking on the appropriate tab and add button.

Refresh will sort the list in alphabetical order.



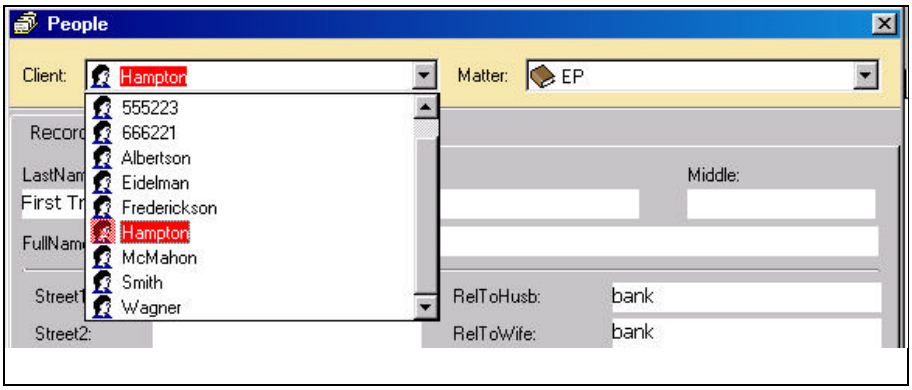
(Implementation Note: It is also possible to have a drop-down box on the WDDATA screen to select the client and matter. Call Eidelman Associates about this customization. In a document management setting, e.g., DOCS Open, it will select files from DOCS Open. Our database will also integrate with the DOCS database or any ODBC database for that matter.)

B. Working with People

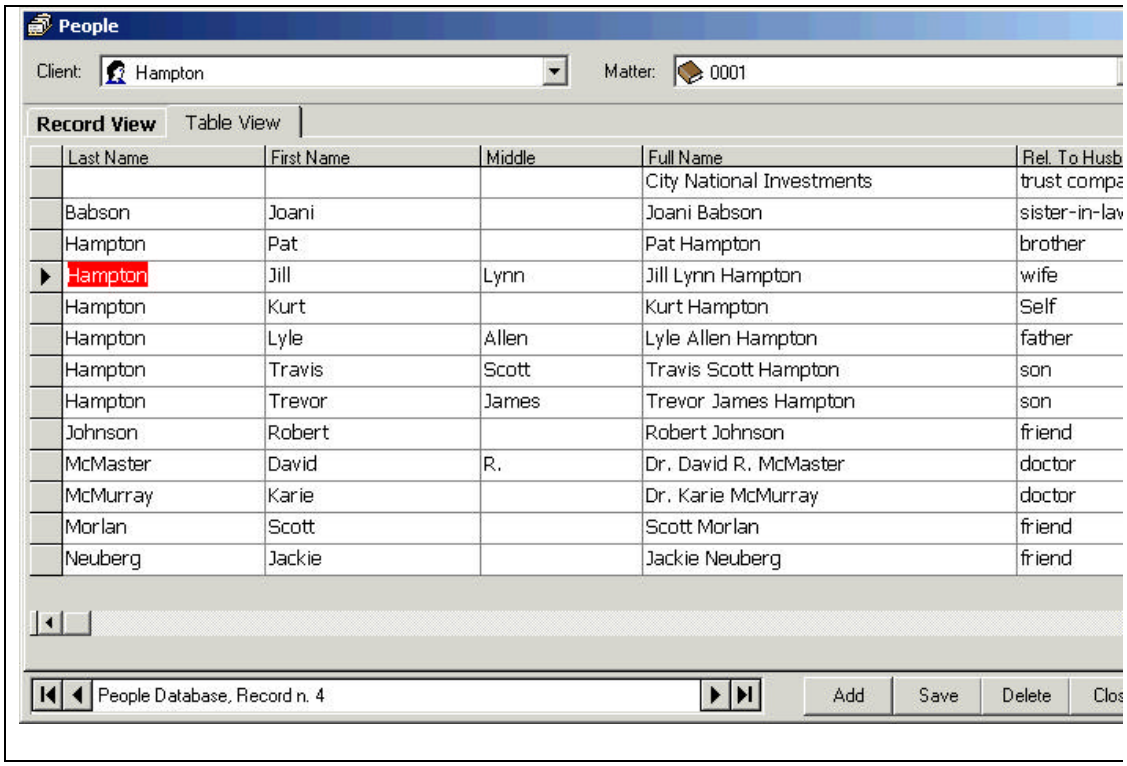
To work with the people, click on the View|People menu, and add as many people as you want for the matter you are working with. You can come back at any time and add more people or make changes.

NOTE: You can filter the data so that you can view ALL the data, or just one client or client/matter.

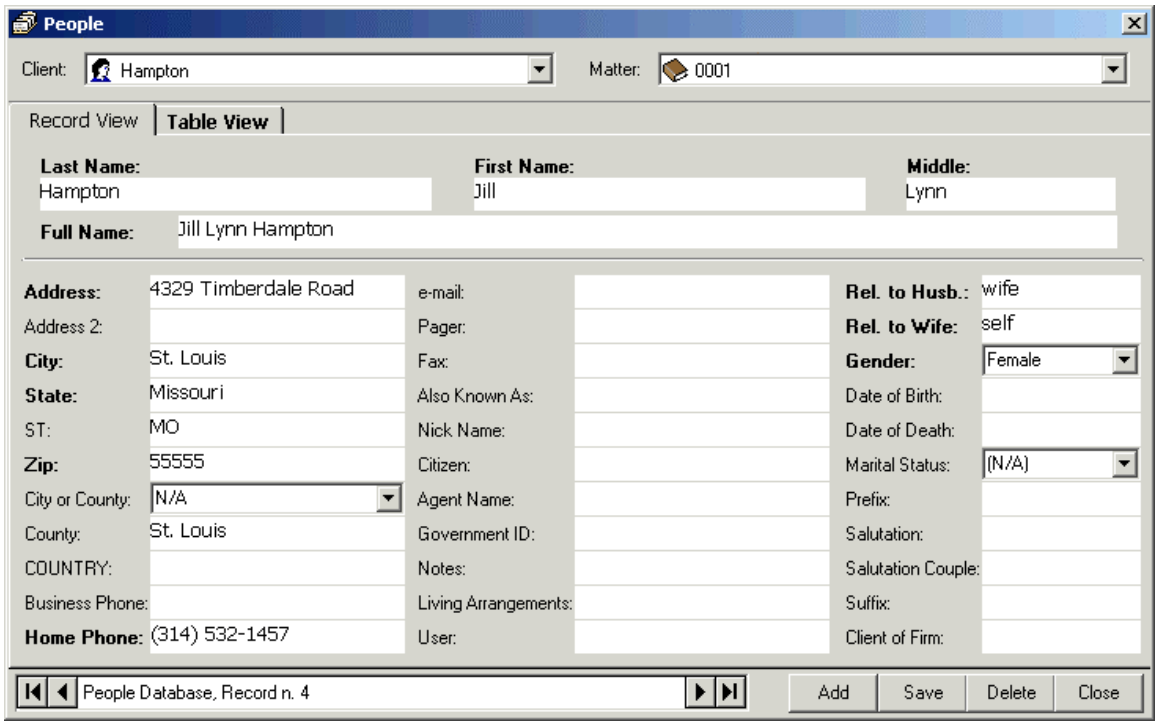
NOTE FURTHER: when working with people in EPPeople, it will display both matters relevant to that Client and people who are assigned to a special matter called “Common.” “Common” is used, for example, to put in the names of attorneys or staff members who frequently serve as either fiduciary or witness.



1. Table View Screen: Enter and view data in table view:



2. Record View Screen: A form for one record at a time



C. Working with the Roles Tree

If you are using the standard WinDraft setup:

When you open an existing WinDraft checklist, it will automatically launch EPPeople and load the right people and roles.

When you create a new WDD file with a new Client.Matter:

- First, go to the File Info screen.
- Second, enter the Client and Matter number into the client and matter fields.
- Third, click the "Assign Roles" button.

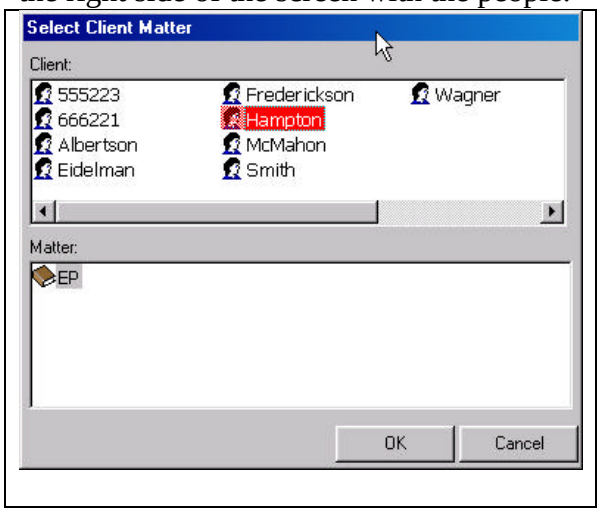
(If the EPPeople interface is in back of the checklist, you may have to click on the EPPeople task on the task bar at the bottom of your screen OR you can alt-TAB to get to the EPPeople screen. You can then add the people to the database, then drag and drop the people to the right roles.

If you launched EPPeople manually, to create a new file from the EPPeople interface, simply run File|New.

1. Select a Client-Matter

(If you haven't already entered the data for the client, matter and people, you should do that first.)

First, select a client and matter to work with. This is required to populate the right side of the screen with the people.

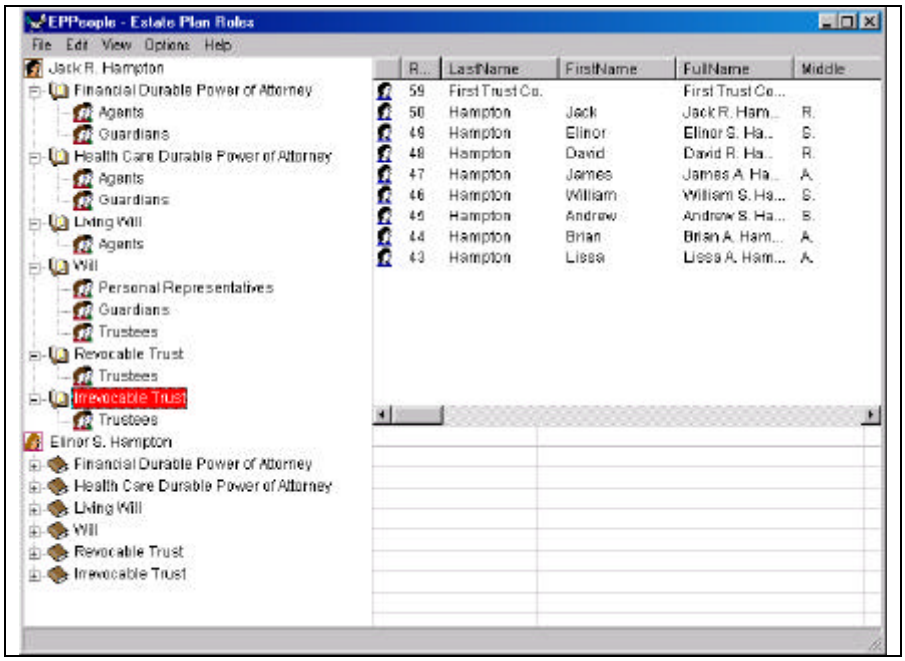


2. The Roles Tree

Here is what the tree looks like before you drag and drop any people to the roles on the left side of the tree control:

As you can see, the husband and wife are at the highest level, then their documents, followed by the roles for those documents.

You can expand and collapse the tree in the same way as with other Windows programs and WinDraft's checklist program.

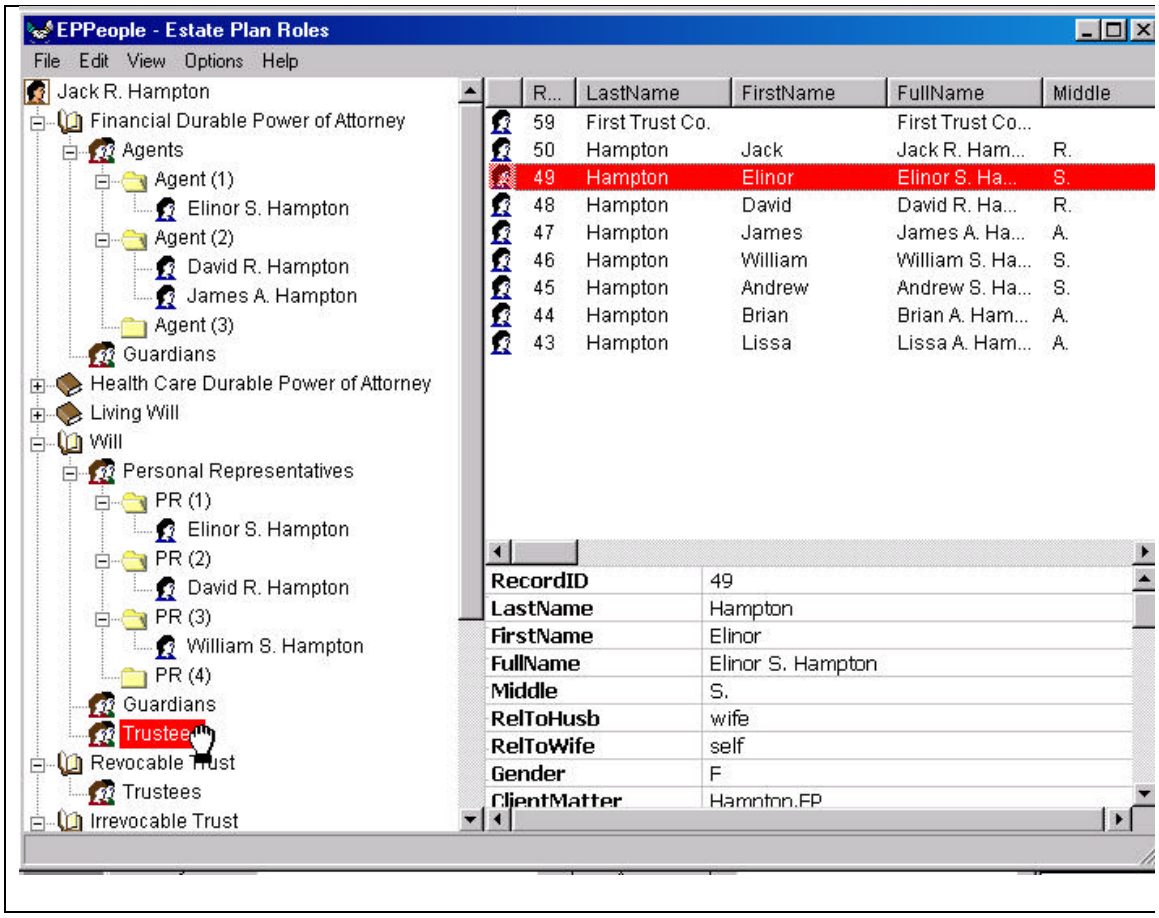


3. Drag and Drop

Start with the Husband's documents. Click on a person on the right, drag them onto the role on the left, and release the mouse button to "drop" the person on the role.

You can have co-fiduciaries by dropping two or more people on any given role.

As you add a person to a role, the system will automatically keep count. That is, if you add Agent(1), the interface will open up Agent(2), and so forth. There is no limit to the number of levels you can have, and there is no limit to the number of people you can have at any level. (Note: The WinDraft System Author must make sure that the document language and the corresponding variables are consistent with the number of levels of roles etc.).



4. Copying Branches of the Tree

a) Copy any Node to any Like Node

The copy function is extremely flexible.

You can copy any node to any node that has the same type of information.

Example:

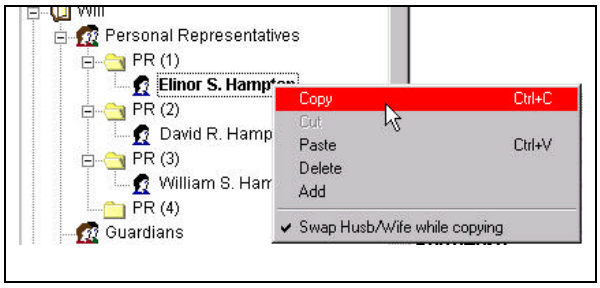
Copy What	To	What gets copied
Husband	Wife	All people, in all roles, for all documents
Husband's Will	Wife's Will	All people and all roles for the will only.
Husband's FDPOA	Husband's HDPOA	All people and all roles for the DPOA

Husband's FDPOA Agents	Wife's Will Personal Reps	All people for the selected roles

NOTE: You cannot currently copy all DPOA people to all Will, because the Will doesn't have agents, and the DPOA doesn't have Personal Representatives. That stated, you can do just about anything else.

Copy and paste work just like other Windows application. You can execute them from either the Edit menu, or by clicking the right mouse button on the node you want to copy from and paste to, like this:

Right Mouse Click:

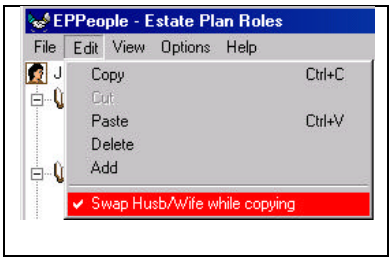


NOTE: If you make a mistake, simply delete the new information you copied.

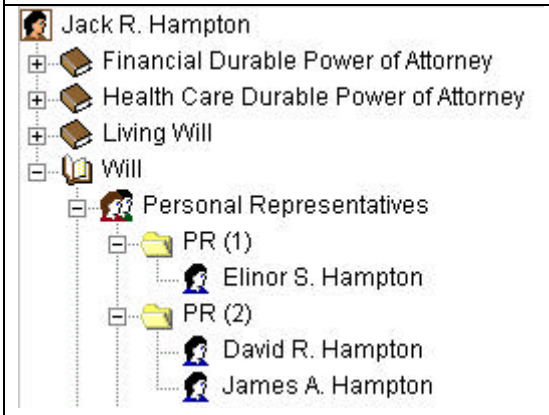
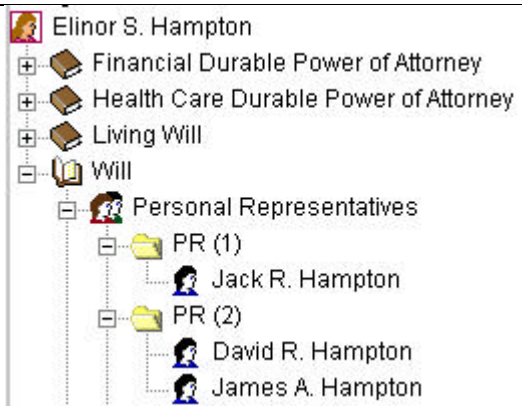
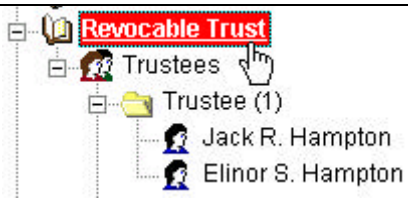
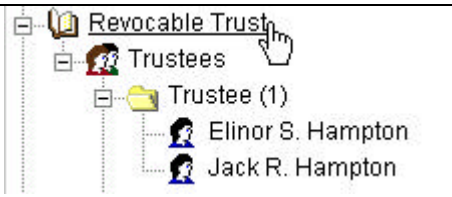
b) Mirror Image Copying

Just as EP Expert offers an elegant way to handle "mirror image" documents for husband and wife, EPPeople offers an elegant way to handle mirror-image fiduciaries when it copies.

When you copy from Husband's roles to Wife's roles, **the system will automatically swap the Husband and Wife roles, pronouns and the order in which they are mentioned in the documents.** All you have to do is check the "swap" option:



Mirror image copy examples:

Copy From Husband	Pasted to Wife
	
	

5. Working with the Tree Display

In general, there is nothing new to learn about working with the tree. It is essentially the same as working with the WinDraft checklist tree or the Windows Explorer tree.

There are a few minor items to note:

1. If you drag a person onto another person, the person you drop will be inserted above the person who was already there.

   But if you drag a person onto a role, the person will be placed at the bottom of the list of people for that role.

   Be careful when doing this, as it will affect how the people are mentioned in the documents.

2. When working with the data, pressing F5 or Edit|Clean up Tree Display will close all nodes that are empty, open nodes that have people in them, and update the display of any data that has changed. This is a helpful cosmetic tool to handle as the last item after entering all people and all roles.

3. When editing data for people and clients/matters, pressing Edit|Refresh People List will refresh the data with any new data that has changed. This particularly applies to a multi-user setting where another user has changed data for a record you have open on the screen.

D. The People List

The people list contains information about the people relevant to a client's file. The colors help distinguish *common people* (in green), *grantors* (in red), and the people filed under the current client (in black).

Common people can be hidden using the **View | Common People** menu.

(Note that the lines they are on remain, but appear blank. Thus, you won't want to use this feature if you have a large number of attorneys, etc.)

By default, people in the list are sorted by last name. However, sorting is possible by any of the columns in the list by clicking on the corresponding column header. It's also possible to rearrange the columns in the list by dragging and dropping the column headers.

Double-clicking on a person in the list takes you directly to the record view of that person.

E. Saving Your Data

WinDraft has two modes to have the WinDraft checklist program save the data. It can either be saved as a separate .EP file, or it can be saved within the standard WinDraft data file – the normal method.

If using EPPeople as a stand-alone program, use File|Save to save the data as a .EP file.

V. Blast-Off to Output Your Data to WinDraft

A. How to Blastoff

In normal operation, EPPeople will blast off automatically when you blastoff from WinDraft's checklist screen.

If the script has not been set up to do that, or if you are operating EPPeople as a separate application, then click the File|Blastoff menu.

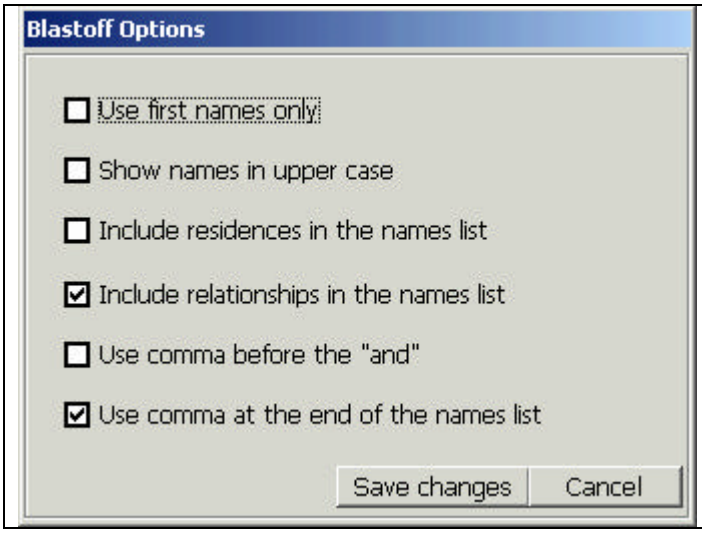
B. What Happens

EPPeople will output a number of temporary data files to the Windows TEMP directory so that WinDraft can read that data as it assembles documents. What those files are and what is in them is described in the Technical Notes section below.

C. Blastoff Options

WinDraft will output the .Names information with various options (See below). Select those options by clicking on the appropriate choices on the "Options" menu.

The options are saved in the WinDraft.ini file, so they may be different if you change machines or login as someone else.


Blastoff Options
Use first names only
Show names in upper case
Include residences in the names list
Include relationships in the names list
Use comma before the "and"
Use comma at the end of the names list
Save changes
Cancel

Mix and match the options!

You can mix and match the options in various combinations.

These options generally speak for themselves, except that the two "comma" options.

- "Use Comma before the "'and'"
  -- Yes – "John Smith, Bob Jones**,** and Fred Jones"
  -- No – "John Smith, Bob Jones and Fred Jones"

- "Use comma at the end of the names list":
  -- Yes – Use this to insert the names in a sentence, such as "I appoint my brother, Robert Smith, to be my Trustee.
  -- No – Use this if the people will be listed in a column, and you don't want the final comma. For example:

  "I appoint the following Trustees, to serve in the order named:

  First, my brother, Robert A. Smith
  Second, my sister, Sally A. Smith
  Third, my uncle, John F. Smith

  Although there are times you might like to mix and match them, at this point the selection is all or nothing.

If you use relationships:

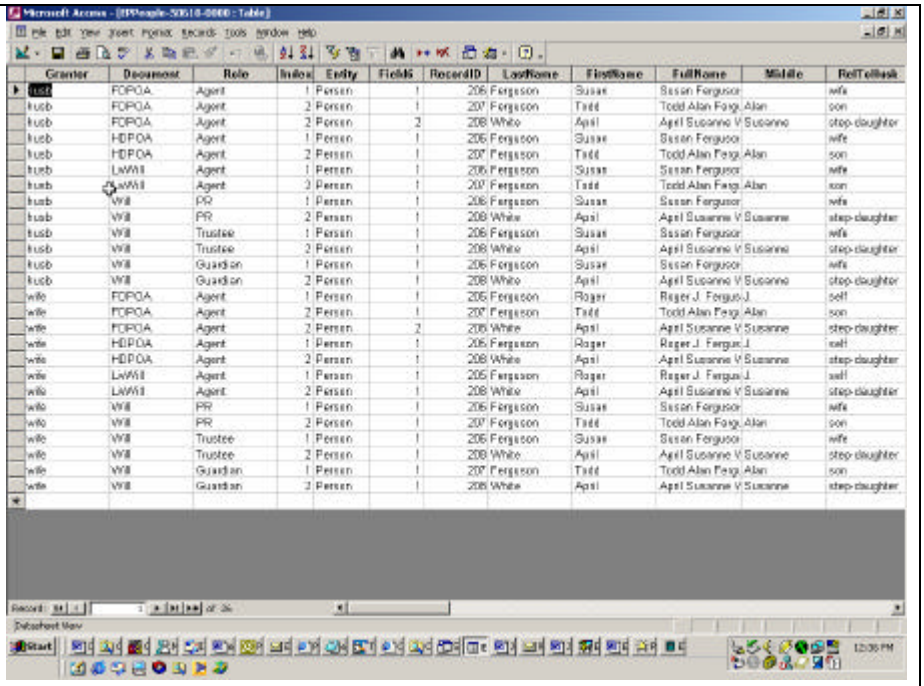
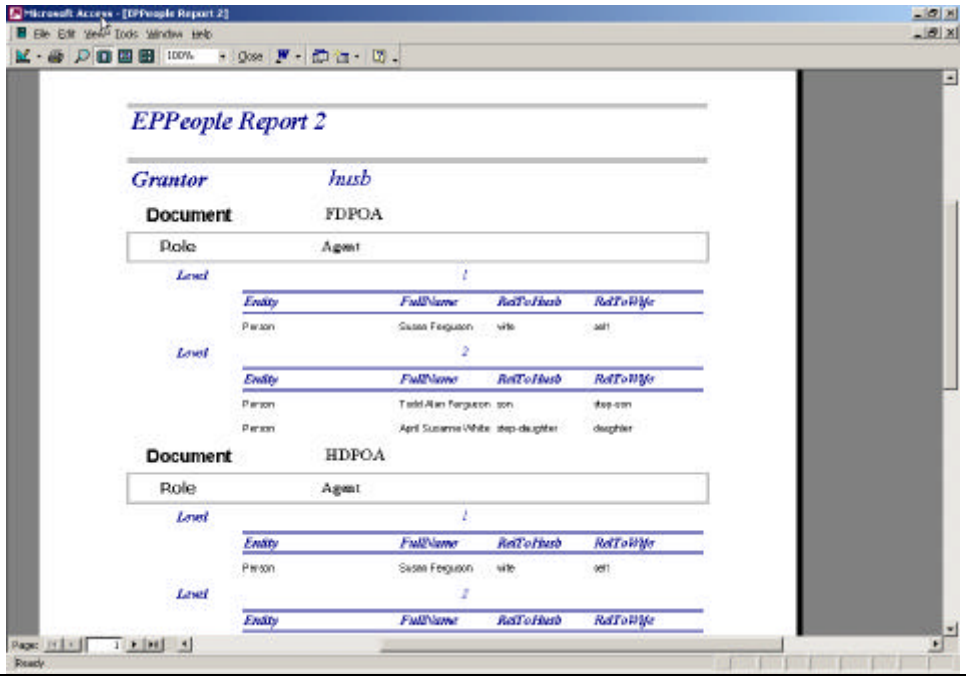
- The system will automatically handle the mirror image relationships correctly. For example, in a second marriage, the husband's would say "my son, John," and the wife's would say "my step-son, John,".
- If the person names him or herself as trustee, it will output "myself" instead of "my self, John A. Smith,".

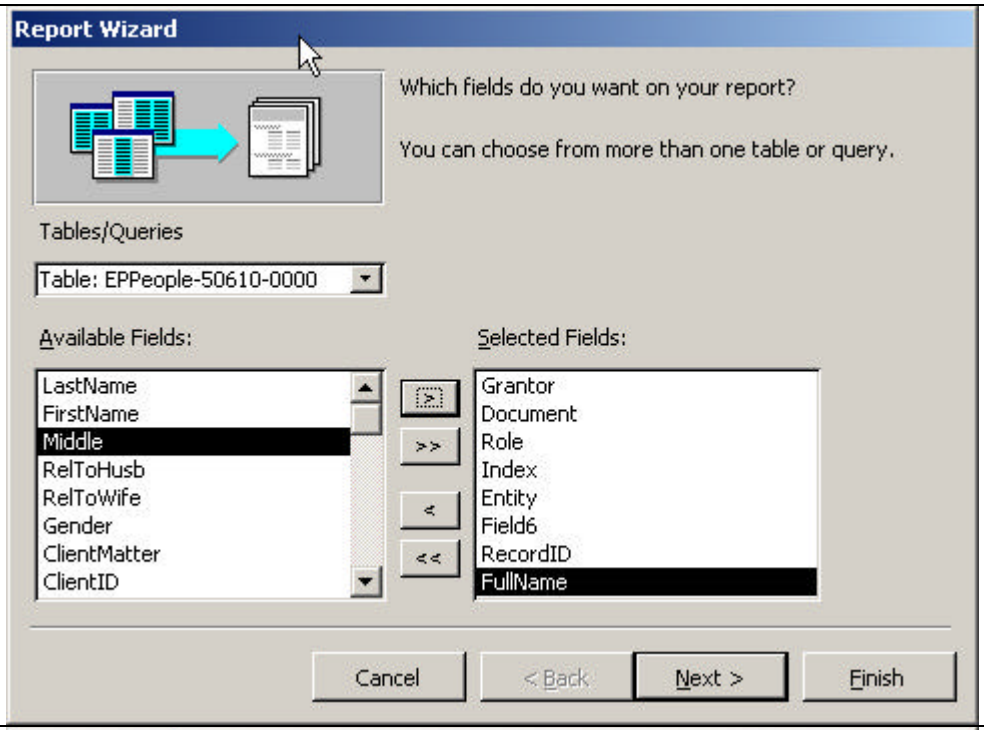
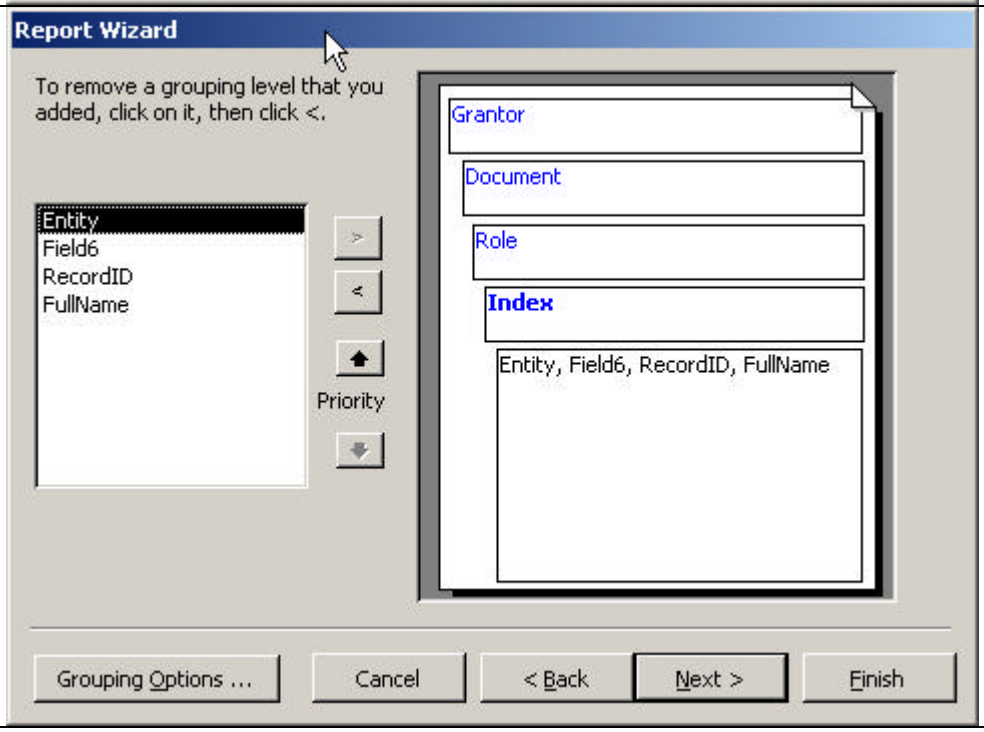
Note that the order will be the same as you see on the tree. If the husband names himself and his wife, it would say "I appoint myself and my wife, Sally A. Smith." If he lists her first, then it would say "I appoint my wife, Sally A. Smith, and myself to be Trustee.

VI. Printing the EPPeople

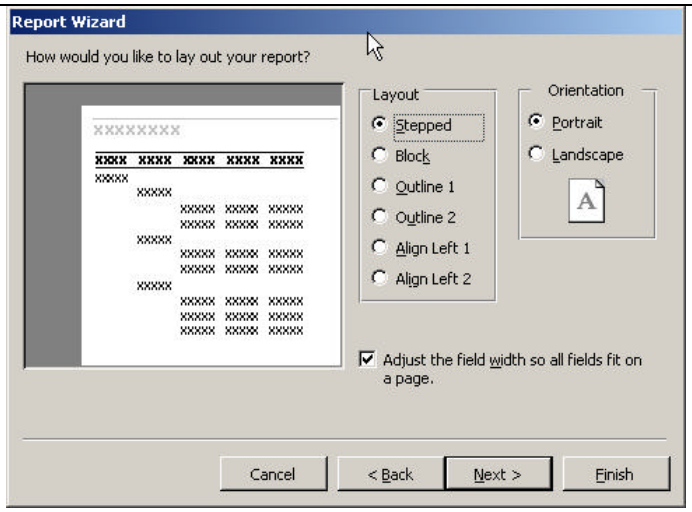
There are now at least 6 ways to print out the data that's in the EPPeople XML file:

A. BASIC PRINT	Version 2.1 and higher has a File|Print menu choice that prints out a basic printout of the people and roles.
B. OUTPUT DATA IN TEXT FILE Data is output automatically to a tab-delimited text file that looks like this when attached	The EPPeople data is available for you to print out or otherwise use the role data. Each time we "blast off," the system outputs not only the data for WinDraft to use in the current set of documents, but an ASCII text file that can be used in any number of different ways. (The format is a tab-delimited table, with the name of the field at the top of each column in the table in the first row.

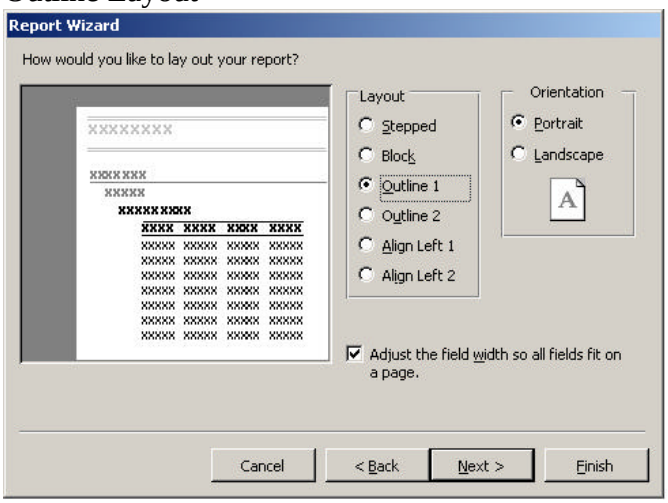
to or imported into an Access file. Right now it goes into the data directory and is named client.matter.txt.	
C. MICROSOFT ACCESS REPORT WIZARD Sample Access Report Created with a few mouse clicks in the report Wizard	

The above was created with a report Wizard that looked like the next few screens.	
	
Layout Options in the Wizard include the following. You can also pick color schemes,	Stepped Layout

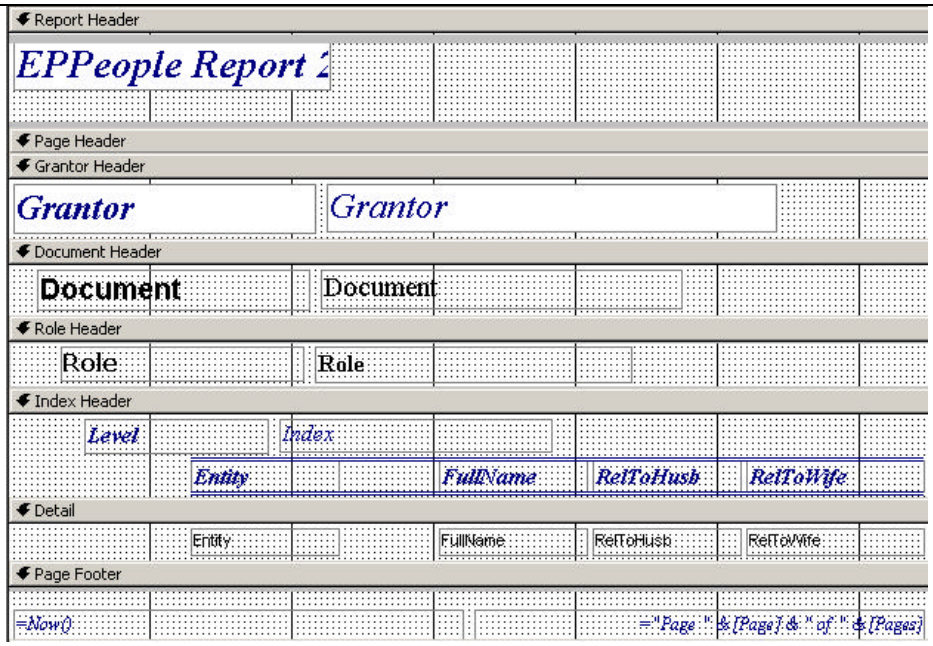
fonts, etc.



Outline Layout



Designer View of the Report created by the Wizard. From here, you can change the text and formatting.



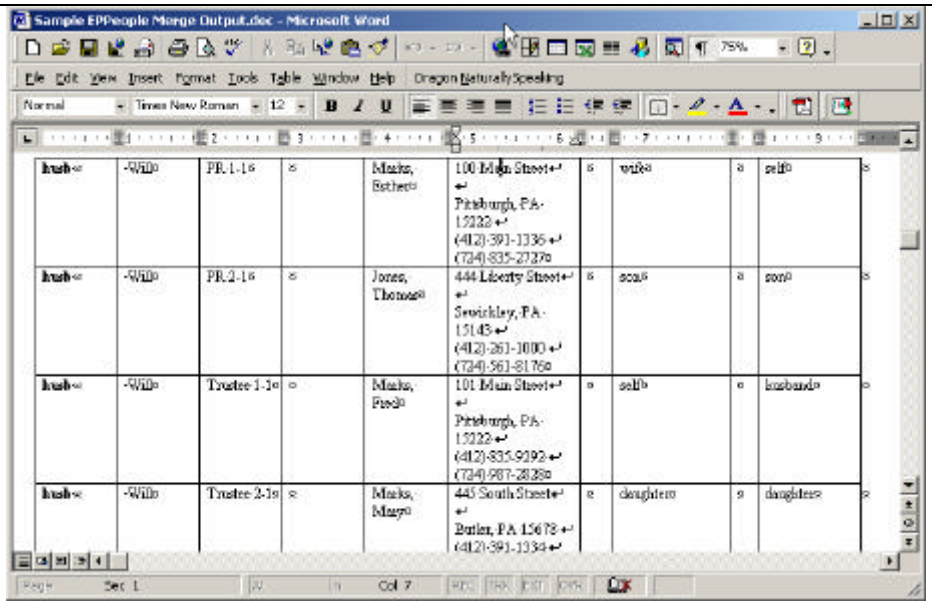
D. EXCEL PIVOT TABLE

The text file can also be read into an Excel table and manipulated in a pivot table that looks like the example to the right.

(Status – We haven't yet created a generic way to create this.)

	A	B	C	D	E	G
9	Grantor	Document	Role	Index	FullName	
10	Husb	Family	Child	1	Trevor James Morgan	
11				2	Travis Scott Morgan	
12		FDPOA	Agent	1	Jill Lynn Morgan	
13				2	Lyle Allen Morgan	
14				3	City National Investments	
15					Robert Johnson	
16				4	Scott Mitchell	
17		HDPOA	Agent	1	Jill Lynn Morgan	
18				2	Joani Burger	
19			Physician	1	Susan S. Edwards, M.D.	
20				2	Jeffrey M. Keightly, M.D.	
21		RT	Beneficiary	1	Trevor James Morgan	
22				2	Travis Scott Morgan	
23			DeathTrustee	1	Jill Lynn Morgan	
24				2	Lyle Allen Morgan	
25				3	City National Investments	
26					Robert Johnson	
27				4	Scott Mitchell	
28			DisabTrustee	1	Jill Lynn Morgan	
29				2	Lyle Allen Morgan	
30				3	City National Investments	
31					Robert Johnson	
32				4	Scott Mitchell	
33			Trustee	1	Jill Lynn Morgan	
34					Kurt Morgan	
35		Will	Guardian	1	Joani Burger	
36			PR	1	Jill Lynn Morgan	
37				2	Lyle Allen Morgan	
38				3	City National Investments	
39					Robert Johnson	
40				4	Scott Mitchell	

Pivot Table 1 | wdoutputall

E. WORD MAILMERGE. Word MailMerge can incorporate the data into Word documents. The repeat loops aren't working quite the way we'd like (there seems to be no way to suppress the output in the columns that should be group fields, such as husb, will, etc. (Arita).	
F. HTML (optionally using XSL) XML/HTML/XSL	

VII. EPPeople Technical Notes

If you are just a WinDraft user, rather than an IS person or system author, you do not need to worry about the following information.

A. Authoring – Programming the Documents

(There are more examples in an appendix. See XIII, Appendix – More Authoring Examples.)

1. Getting the Data into the Documents using GETDATA

With the EPPeople program, the beginning of each document template will generally have two GETDATA command lines to get these temporary "blastoff files." Other GETDATA commands are in EPStart.doc or GRGender.doc on the subdocs directory. Those GETDATA commands are:

- Estate Planning questions other than fiduciaries.
- Fiduciaries – all (with long paths for variable names – e.g. (Husb.Will.Trustee(1).Name)

- Fiduciaries – husb or wife for all documents (medium length paths for variable names – Will.Trustee(1).Name
- Fiduciaries – each document (shortest length paths for variable names – Trustee(1).Name.

These files are automatically written to and read from the Windows TEMP directory, which we get from the Windows API.

2. "Scope" of the Variables

Each variable for a person or a role level can be referred with or without the "path" that shows what document and which grantor it relates to.

For improved readability and less risk of errors, we recommend that you refer to a variable by using the shortest version of the name. (Note: If you *want* to use the longer name, you *can*. It does no harm. It's just that the shorter names are easier to understand when you are reading the document.)

Example of how to use the variable for Initial Trustees in the husband's revocable trust (at the Role Level):

Example – If var is used in this document	**Example "Home" document of the var.**	**Variable as used in the document– Initial Trustee(s) in the Husband's Revocable Trust**	**GETDATA reads data from this temp text file**	**Explanation**
Husband's Rev Trust	Husb's Rev Trust	**[Trustee(1).Names]**	husb.will.txt	**Scope: Only in the "home" document.** If a var is in its "home" document, no qualifying path is required. You can optionally use a longer path name, but it is better style not to, both for readability and portability.
Husband's Pour-over Will	Husb's Rev Trust	**[RT.Trustee(1).Names]**	husb.txt	**Scope: In any of husb's docs**. Because the *will* is calling a *trust* variable, you need to put "*RT*." in front of it. But you don't need the "husb." because it is the husb's doc calling another document of the husb.
Letter to both clients summarizing the documents.	Husb's Rev Trust	**[Husb.RT.Trustee(1).Names]**	wdoutputall.txt	**Scope: Any document.** You need the full path because it's being called from a document that is not only not the "home" document, but not even the husband's document.

Example with Same Variables down to the Person Level:
Note: When referring to a person, rather than a role, in the examples above, the use of the Husb. or Wife. and document. to the left of the role is the same as the example above, but you replace the ".Names" with ".Person(1).Name", etc.

Example – If var is used in this document		**Variable – First Person serving as Initial Trustee or co-trustee in the Husband's Revocable Trust**	

Husband's Rev Trust		**[Trustee(1).Person(1).Name]**	
Husband's Pour-over Will		**[RT.Trustee(1).Person(1).Name]**	
Letter to both clients summarizing the documents.		**[Husb.RT.Trustee(1).Person(1).Name]**	

3. Using Personal Pronoun Functions

a) How the functions work

EP Expert uses a gender "function" that works like this:

> I give to [HimHer(GrantorSpouse)], to be [HisHers(GrantorSpouse)]…

If [GrantorSpouse] in this example = 1 (for male), the sentence will output

> I give to him, to be his…

If [GrantorSpouse] in this example = 2 (for female), the output will be:

> I give to her, to be hers…

To see how these variables work, see GenderTerms.RUL, an ASCII file on the practice system subdocs directory. It identifies every gender-specific and number-specific term that has been set up, and you can add your own.

b) Using the functions with EPPeople

When using the output from EPPeople:

When working with an individual person, what goes in the parentheses is the index variable for that role.person, such as:

For each person, there is a name, state, zip, etc., and there is always a .index, which will have the following values:

Value of .index	Gender and number	[HeShe(person(1).index)]
1	One male	he
2	One female	she
3	Multiple people	they

4	One entity (corp, etc.)	it

JAE NOTE: Originally, we used to always use ".index". However, now EPPeople will also output the root variable with no "dot" that is the same as .index. This makes the appearance of functions much easier to read, such as [HeShe(Trustee(1))] instead of [HeShe(Trustee(1).index)]. The use of ".index" is now optional.

To the extent that other parts of the manual discuss or give examples using the .index, they apply equally to the root variable without the .index.

The ".index" will be output both for each role.person and each row.level (eg. Initial trustee, first successor, etc,) and the number of each person who may act as cotrustee at any level. Thus, you can use either or both of the following:

"[Trustee(1).Names] shall serve as Initial Trustee."

"[Trustee(1).Name] shall serve as Initial Trustee."

- Note that the above has one .Name/.Names for each level of succession of a role. This is how you refer to the one or more people in a role collectively in the middle of a sentence. The role(level) variables, such as [Trustee(1).Name], are for the 1 to n people in that role, collectively, who may be a fiduciary or co-fiduciaries.

  You cannot use this to insert names individually into a document, such as in a column list or in beneath a signature line. For that, see the example below.

- The author can use either "Name" or "Names" interchangeably in the template.)

Here's how you use the pronoun functions:	
Desired result:	If he shall survive …
WinDraft coding:	If [HeShe(Trustee(1))] shall survive…

"I appoint [Trustee(1).Person(1).Name], who lives at [Trustee(1).Person(1).Street], [Trustee(1).Person(1).City].

- *Note that here there is one variable for each PERSON. By going to the person level, you deal with each person individually. You would use this form of the variables when working with one person at a time, such as inserting their names beneath signature lines.*

4. Using Word fields

This document is not to explain how to use fields.[3]

You can embed a field with a set command near the beginning of the document.

> Note: You can avoid the "bad cosmetics" of the "Error. Bookmark Not Found" message by setting the Word field variable near the top of the document. When you then recalcualte (Cntrl A, Cntrl F9) this will eliminate the "Error" message. If you do not SET the Word field near the top of the document, you WILL get the "Error" message, but when the document merges the value will be correct. Therefore, this is a cosmetics issue.

When WinDraft runs, it will replace the WinDraft variable so that the value of the WinDraft variable is used by Word in recalculating Word fields. For example:

{SET Agent1HeShe "[HeShe(Agent(1))]" }

This will allow you to insert the field {Agent1HeShe} anywhere in the document and have it insert the WinDraft variable.
We do this frequently with the grantor and grantor's spouse variables, and with the first level fiduciaries, such as agent in DPOA's, (Durable Power

[3] You can read about that elsewhere, e.g. Insert, Field in Word, then click Help. This will get you started.

of Attorney), but not with variables that are only used one or two times in a document.

5. Using the Fiduciary Variables in Context

a) Without FOR Loops

Simply insert them in a sentence wherever they go. Such as, "I appoint [Trustee(1).Names] to serve as co-Trustees."

b) With FOR Loops

The sequences of levels represented by the number in parentheses allow them to be used in a FOR loop. See WinDraft author's manual for how FOR loops work.

An Example that shows the syntax is as follows:

I appoint the following Trustees, who shall serve in the order listed below:

[;; the next line will insert a line for each trustee:]

[FOR I = 1 TO Trustee.Levels INCLUDE "TrusteeName.doc"]

(In this example, TrusteeName.doc includes one line, like this:

[;; TrusteeName.doc]

[Trustee(I).Name]

c) Combining The Two Approaches

Here's an example that uses the two approaches, depending on whether or not there is just one trustee:

[IF Trustee.Levels = "1"]

I appoint [Trustee(1).Names] to serve as Trustee.

[ELSE]

I appoint the following Trustees, who shall serve in the order listed below:

[;; the next line will insert a line for each trustee:]

[FOR I = 1 TO Trustee.Levels INCLUDE "TrusteeName.doc"]

[ENDIF]

B. Authoring – Programming the Checklist

1. Use VBScript or JavaScript programming in a .WDS file to have variables and Events in WinDraft's checklist interact dynamically with EPPeople.EXE.

2. Events that interact with EPPeople can include Windraft system events (Blastoff, Save, Open), or can be programmed to react when the user exits from a field or clicks an Author-defined button.

3. You can also interact with EPPeople's Database or XML files from other parts of the system, such as blastoff programs in Word VBA, etc.

C. Authoring – Editing Templates

1. Understanding EPPeople Templates and Data Files

When you create a new data file that contains EPPeople data, that data is stored in XML, and it is based on an XML file stored on ...\template with names like "General.ept" (for married husband and wife), "wifeonly.ept", etc.

If you want to change the relevant grantors (husb, wife or both), documents, or roles, you do that the same way in a client data file and in the templates.

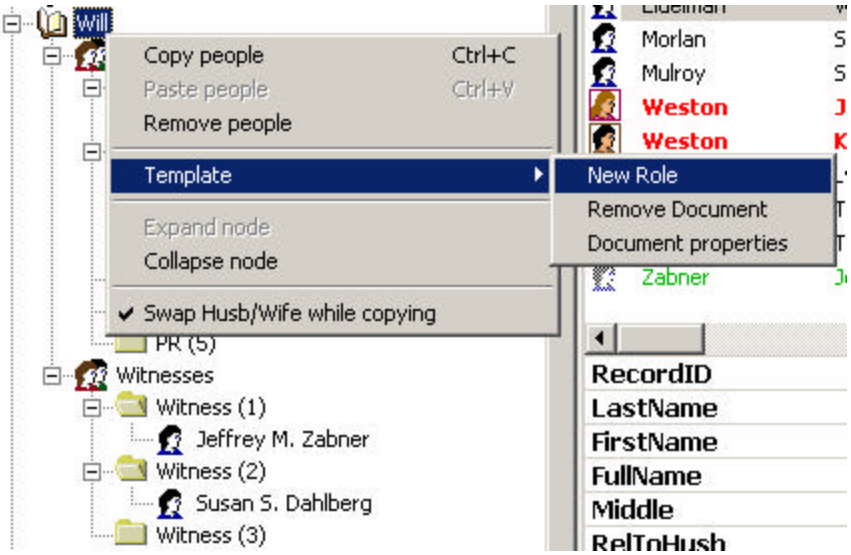
a) First, open the data file or template file.

b) Second, right mouse click on the appropriate level and make the changes you want.

c) Finally, save the data file or template.

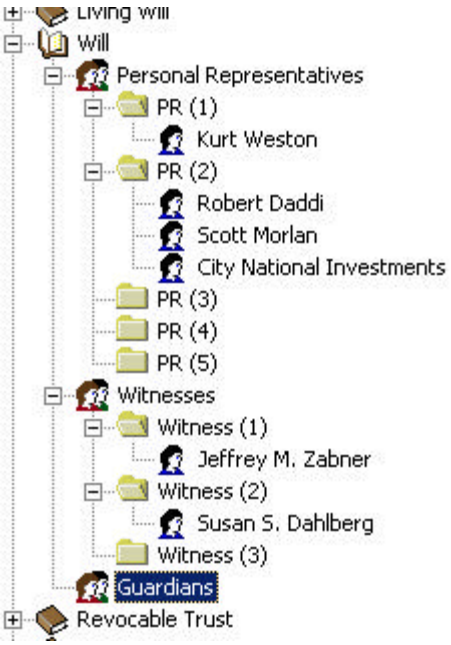
2. How to create a new role or document

To create a new role or document, right-click on the node hosting the new item and select **Template → New *node*** from the popup menu, where *node* is either a document or a role:



Enter the name of the new node in *Text to display* and the corresponding blastoff code in *Blastoff Text*.

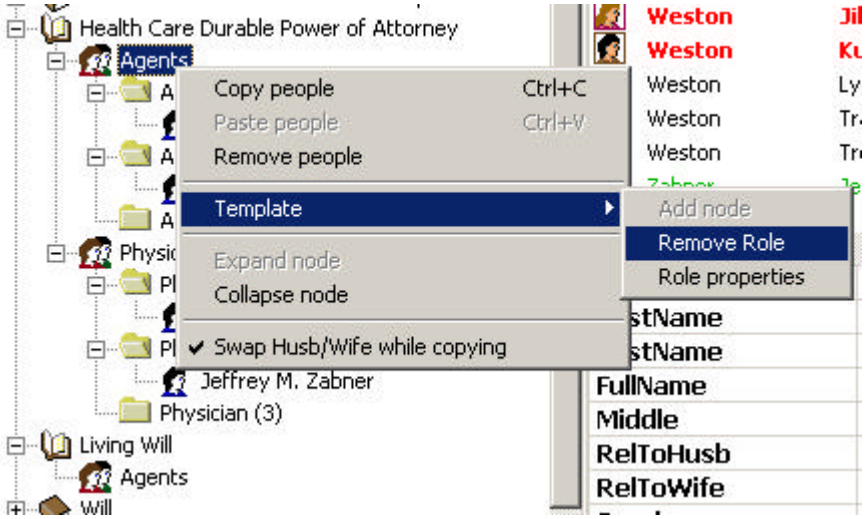
In example above, after clicking on *Create Role*, a new role will be added to the *Will* document, as shown below.



Only documents and roles can be added to the template. Levels are automatically added when dragging people onto a role or onto the last level of a role. Grantors cannot be added although they can be removed from the template, as explained in the following section.

3. How to remove a role, document or a grantor

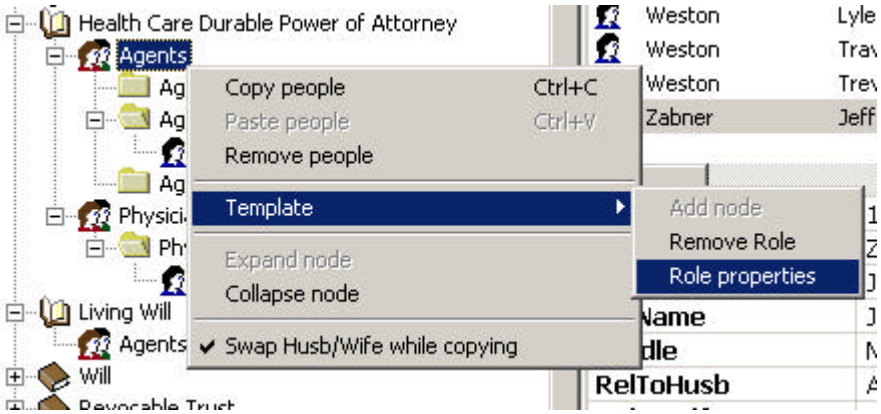
Right-click on the node that you want to remove and select **Template** > **Remove *node***, where *node* can either be a document, a role or a grantor.



By removing a grantor, you can create templates for *Man-Only* or *Woman-Only* estate planning.

4. How to change the properties of a node

Right-click on the document or role that you want to modify and select **Template > *Node* properties**, where *node* is either a document or role.



Change the properties of the node in the following dialog box, and click on *Apply*.

The *Node properties* menu applies only to roles and documents.

5. How to reorganize the tree structure

It's possible to rearrange roles and documents in the template by drag & drop. When dragging roles or documents, you need to keep the SHIFT key pressed. This is to distinguish the Drag and Drop operation on template nodes from the Drag and Drop on people, and to avoid accidental rearrangements of the tree structure.

6. Summary

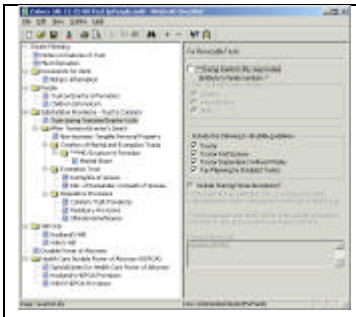
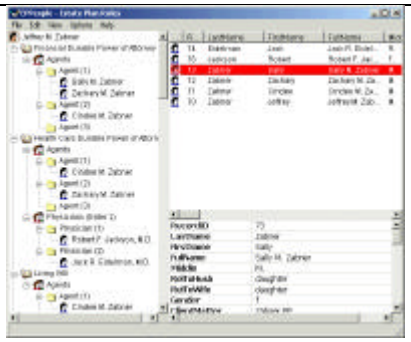
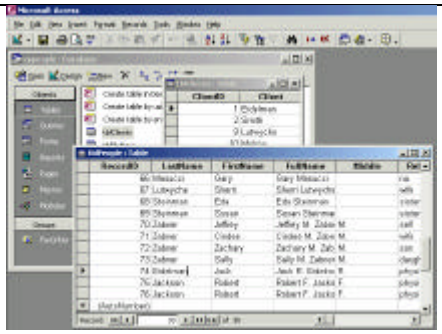
These are the template operations available for each type of node in the tree:

Person	N/A
Level	N/A
Role	Remove role Role properties
Document	New role Remove document Document properties
Grantor	New document Remove grantor

D. "Under The Hood" -- Working with EPPeople Role Files

1. How the Files Fit Together

Under the new system, a complete set of data consists of 3 types of data files that "talk to each other."

	<- >		<- >	
WinDraft .WDD file. Contains answers about the estate plan. One per client.		**EPPeople .EP file (usually embedded within the .WDD data file)** XML file that contains the links between documents, roles, and people. The links to people point to records in the database file.		**Microsoft Access .MDB.** Database file that contains information about people, clients, matters. (You don't need Access to work with these files.)

a) About XML

XML (Extensible Markup Language) is a format for data that is taking the computing world and the Internet by storm. It is a hierarchical format (like an outline) that stores data in an ASCII text file that you can edit with Notepad, Word, or specialized editors like XMLSpy and EditML.

What's so good about XML is that it is a format that, unlike most other formats, it can be read both by humans and by computers.

See "Get Up to Speed with XML" (http://www.xmlmag.com/upload/free/features/xml/1999/01win99/bfwin99/bfwin99.asp) and "Why I like XML" (http://www.xmlmag.com/upload/free/features/xml/2000/04fal00/dw0004/dw0004.asp). Also see www.xml.org.

EPPeople stores information about the fiduciary roles and other roles in an XML format.

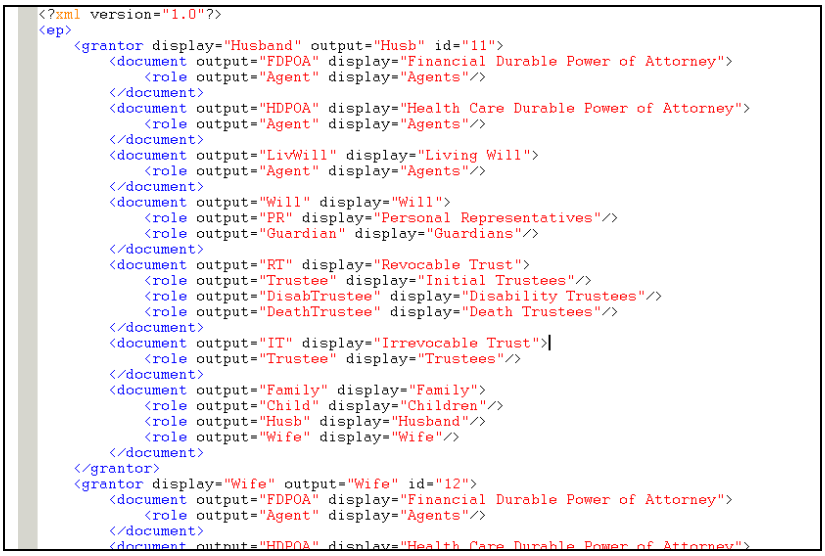
b) *General.EPT*

If you want to change the grantor, documents or roles in a client data file or a template for a new one, it's easy. Just right click on the grantor, document, or role, then click "Template", and select the appropriate action.

For single people, there are template files called: "ManOnly.EPT" and "WomanOnly.EPT".

When you create a new data file, the system creates a new file that defines the documents, roles, and output information. It does so based on a "template" file called "General.ept" or another template file that has the file extension ".EPT". The EPT files are stored on a directory set in WinDraft.ini, which by convention is the practice system \Template directory.

The content and format of a General.EPT file looks like this:



```
<?xml version="1.0"?>
<ep>
    <grantor display="Husband" output="Husb" id="11">
        <document output="FDPOA" display="Financial Durable Power of Attorney">
            <role output="Agent" display="Agents"/>
        </document>
        <document output="HDPOA" display="Health Care Durable Power of Attorney">
            <role output="Agent" display="Agents"/>
        </document>
        <document output="LivWill" display="Living Will">
            <role output="Agent" display="Agents"/>
        </document>
        <document output="Will" display="Will">
            <role output="PR" display="Personal Representatives"/>
            <role output="Guardian" display="Guardians"/>
        </document>
        <document output="RT" display="Revocable Trust">
            <role output="Trustee" display="Initial Trustees"/>
            <role output="DisabTrustee" display="Disability Trustees"/>
            <role output="DeathTrustee" display="Death Trustees"/>
        </document>
        <document output="IT" display="Irrevocable Trust">
            <role output="Trustee" display="Trustees"/>
        </document>
        <document output="Family" display="Family">
            <role output="Child" display="Children"/>
            <role output="Husb" display="Husband"/>
            <role output="Wife" display="Wife"/>
        </document>
    </grantor>
    <grantor display="Wife" output="Wife" id="12">
        <document output="FDPOA" display="Financial Durable Power of Attorney">
            <role output="Agent" display="Agents"/>
        </document>
        <document output="HDPOA" display="Health Care Durable Power of Attorney">
```

As you can see, the General.EPT file defines the documents, roles and output, but has no information in it about people.

There can be different .EPT templates for different firms, states and factual situations. By default they are named General.EPT, but they can also have other names, and a macro can use different EPT files to create a new .EP roles file depending on the facts.

c) *Client .EP files:*

As mentioned above, EPPeople has two modes of operation:

- Normal: Integrated so that EPPeople data is saved with the WDDATA Checklist, so there are no separate EPPeople data files.
- Special: Launched by itself from Windows.

Except for importing and exporting, in normal operation the user does not need to worry about .EP files. WinDraft "stuffs" them into the normal .WDD WinDraft checklist data file and manages them automatically.

The following was written before we started storing them in the WDD file. Everything below except for File operations is still valid.

The EPPeople role files are formatted in XML, and have file names that end with ".EP". They are generally stored on the practice system data directory, such as \windraft\epFirm\data. They look like the following example:

(JAE Note: The XML file now has the person's name in it as well, which will synchronize the info if the person id has changed in the database. (This would occur, for example, if the data were entered in one MDB, then imported into another.) JAE: Please paste in a new picture.)

Note that the file is the same as the EPT template file, except that it contains additional information about each person and about the levels for each fiduciary role.

Because the person's name, address, and other personal information is looked up in a database, the EP file does not contain the actual information about each person, but rather an index field that is a link to the person's information in the database.

d) Technical Notes on Working with the XML and MDB files

This approach works great, as long as the environment is stable, in that:

- **The database must match the .EP data file**. If you email an .EP file to someone who is set up with a different database, or different version of the same database, the "person ID" field will point to the wrong person, and it won't work right.

  (At the moment, there is no way to combine 2 different database files unless they started from the same one and are replicas that can be sync'd using MS synchronization technology. This is what a firm will use to share network databases and off-line databases used on laptops.)

- **The roles defined in the .EP file must match the roles used in the documents**. If an .EP file has roles and people in it, and we create a new role by adding it to the documents and NEW.EPT, the new role will be picked up

by new .EP files created from that point on, but the new role won't automatically be added to the existing .EP files.

(Programming note: This will be changed in a future version that will be released in January, 2001.)

Neither of these is a problem in a production setting, but this can be inconvenient for a developmental, testing environment in which you want to email sample data files and databases around. For testing, until you get it the way you want it, it is best to be as flexible as possible.

(Programming Note: Should we eliminate the need for the MDB file and store all the data in the XML file? We'll have to see how much the users see the MDB as an advantage vs. something that gets in the way. It's much better in a stable, network environment, but a pain if users are emailing data files around and don't have enough tech support to handle database synchronization.)

VIII. New Features Not Yet Documented

This is a list of features that have been added to the program, but not yet documented:

- More robust output of children information:
  - His, hers, ours
  - Step children, natural/adopted children, all children
  - .ChildOf

- .TypeCode output
  (Each role has a variable .TypeCode, such as [Trustee(1).TypeCode]. It is a text string that contains one or more characters: G for Grantor, S for Spouse, O for at least one other real person, and C for Corporation.

- Using vbscript to pull beneficiaries and children into the WinDraft checklist

  - (Add Husb & wife info into checklist)

    It is possible to set up a button or other event in the WinDraft checklist so that when you click it, fields from the roles table are imported into the Checklist.

    For example, you can set up a FOR loop in the checklist for beneficiaries, with questions about the trusts being set up for each of them, and a Role in

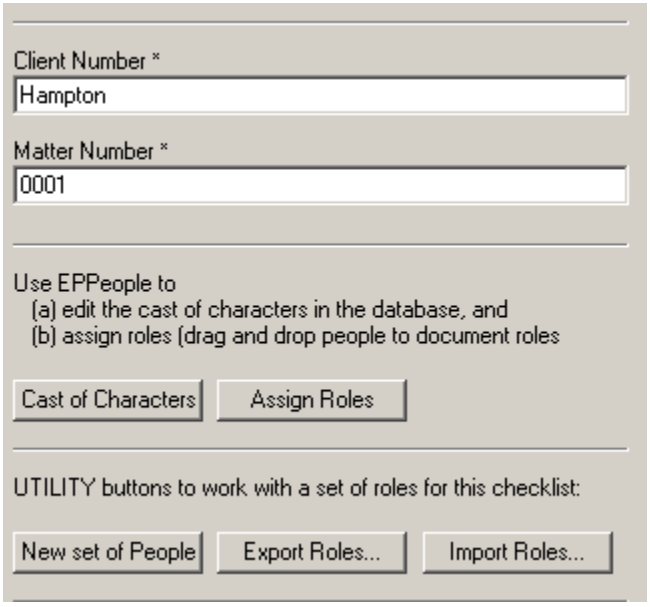
the EPPeople tree called “Beneficiaries” for the Revocable Trust. When you click the button, the people will be pulled in from the tree into the checklist. You can then answer the questions, blast off, and get a separate trust with correct names and other info from both the checklist and EPPeople merged into the document. Call EA if you are interested in an example of this.

- Husb. and Wife. variables for mirror image. (Move from WDOpen.dot into WinDraft.dot)
- **Editing the template, moving nodes. Also can change General.ept this way.**
- **Sorting, moving columns**
- **Double Click on person in roles tree to edit a person.**

Explain – Switch to, Retry messages with export, etc.

Explain setup isssues -- Explain about self-registering and our Wise update.

New WDS buttons:


Client Number *
Hampton
Matter Number *
0001
Use EPPeople to
(a) edit the cast of characters in the database, and
(b) assign roles (drag and drop people to document roles
Cast of Characters
Assign Roles
UTILITY buttons to work with a set of roles for this checklist:
New set of People
Export Roles...
Import Roles...

IX. Wish List

Tooltips in data entry.

X. Appendix – Setup and Troubleshooting

A. Setup and Architecture

Setup depends upon the configuration of the firm's network, etc. Generally it can be described as follows:

1. 3 Parts of Setup (These may be combined into one or more setup routines.

a) Set up DLL's and registry settings to make the Windows installation compatible with all of tools being used. (All Microsoft. They are latest versions of ADO, XML, etc.)

b) Set up Application files you need to run the EPPeople part of the application. They go on Drive C: if you are a typical local user, and on a network "share" (a shared directory) if you are working on a network.

c) Programming or setting up the practice system variables and WinDraft scripting to "lock into" the EPPeople way of doing it, including stylistic and legal drafting issues.

2. Troubleshooting: Temp Blastoff Text Files on Temp Directory

a) Temp File Output Location.

Where are the temp files?

The temp files are on your Windows temp directory, which varies depending on your setup.

The different modules of WinDraft pass data to each other by writing the data to temporary files, which we call "Blastoff files." They are in an ASCII text format and can be edited with Notepad, Word or your favorite editor.

In earlier versions of WinDraft, the temp directory was set in the WinDraft.ini file.

Current versions of WinDraft rely on Windows, and Windows uses the temp directory. Typical settings are:

Operating Environment		Default Settings
Windows 3.1, NT, 2000		\temp
Windows 95/98		\windows\temp\
NT, Win2K, XP Profiles		C:\Documents and Settings\jae-nt5\Local Settings\Temp

Here is a directory display of 2 of the files in my temp directory in my jae-nt5 login profile:

Husb.FDPOA....	C:\Documents and Settings\jae-nt5\Local Settings\Temp	2 KB	Text Document	12/9/2000 7:50 PM
Husb.txt	C:\Documents and Settings\jae-nt5\Local Settings\Temp	2 KB	Text Document	12/9/2000 7:50 PM

(The temp directory can be obtained programmatically from a Windows System Object in VB, Windows API in VB or C++, and in batch files using the TMP or TEMP environment variable.)

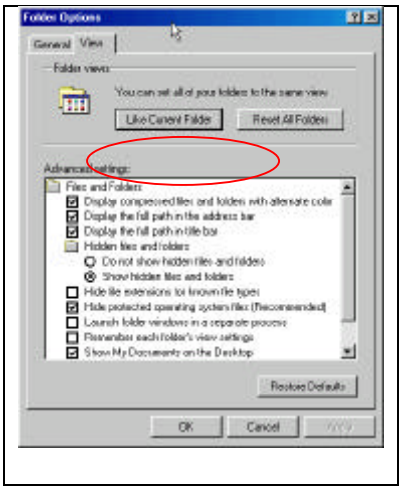
b) Making your Temp files Visible

Q. I can't find the temp files, even by searching my entire hard disk. Am I in the twilight zone?

A. No, you aren't in the twilight zone. The files are hidden from your display.

If you want to "go under the hood" and look at the temporary blastoff files and you can't find them:

I have found that on a Win 2K machine, and some other environments, searching for the temp files and trying to find them by navigating the \documents and settings folder that contains the profile, comes up blank, and I've gotten very frustrated that I knew they were there but couldn't find them. The trick is that you have change your file settings because they are hidden away. (Change them on the Tools|Folder Options menu.)



B. Note on Database Connections:

WinDraft.ini practice system now has the following items:

EPConnectionString=

See http://www.able-consulting.com/ADO_Conn.htm#DSN for examples of the kinds of entries that can be put in here. (There are links from there to Microsoft documentation on the various options.) This setting is used to make EPPeople compatible with a people database that is in a wide variety of formats, connected using a variety of methods and drivers. Typical entries would be:

EPConnectionString=DSN=EP People
EPConnectionString=Dbq=n:\windraft\EPFirmIntitials\data\eppeople.mdb

C. Troubleshooting EPPeople Issues – Following the data

We may hear, "I added three additional people, but they didn't stick. They didn't come into the document." Troubleshooting this is difficult unless we know the specifics. Here are the things to look for:

1. Database Entries

Are the people you entered in the database? If you add a person, that person should still be in the database, whether or not you dragged any of the people into roles or saved the WinDraft data file. (Take a look with EPPeople's view people menu choice, or from the data checklist by clicking on the "Edit People" button. Or you can go into the EPPeople.mdb file with Access and take a look.)

Is the client code field right? Note that the client number in the client database and the EPPeople XML data must match. Generally, this must also match the Client number in the checklist. Please don't change client numbers unless you are sure you know what you are doing or you may get unpredictable results.

2. The Roles Tree

If you dragged and dropped the people into the roles tree, are they still there? Don't forget that if you add them to the roles tree, you still have to Save within the WinDraft checklist in order for the role data to be saved.

3. "Blast-off" Temp Text Files

When you "blast off," EPPeople will output a number of text files to your Windows TEMP directory. (If you don't know how to find your temp dir, search for one of the files. They have file names like husb.txt, husb.rt.txt, etc. See Troubleshooting: Temp Blastoff Text Files on Temp Directory, below.) Open these files and see if the data is in there. If the data is in the roles tree, and if EPPeople is operating correctly, then blast-off should output the data to these files.

4. Variables Merged into the Documents

If the data is in the temporary blastoff files, but not coming into the documents, then there may be a problem with the GETDATA commands in the model documents. GETDATA commands to pull in the right data are in the model documents, generally organized like this:

Variables for which the document is the "home" document.	DOT files being merged (eg. FDPOAHusb.dot, RT_Wife.dot, etc.)
Variables for all documents	..\subdocs\EPStart.Doc
Variables for all of man's documents or all of woman's documents.	..\subdocs\GRGender.doc or EPStart.doc

D. Setup Procedures on Windows 95 and NT

EPPeople Setup Notes

Loading Windows Components that EPPeople Uses

Most files and setup notes are on http://eahome.lawtech.com:85/~clifiles/eppeople2/

JAE 11/12/2001

A. **Table of components** you need to load for each Windows environment:

(As you can see, if you have Windows XP, you don't have to load anything except our setup file. If you have Windows 95, you need to load everything.)

	Win 95	**Win 98/ ME**	**Win NT**	**Win2000** (Ships with IE 5.0)	**Win XP** (Ships with IE 6.0)
DCOM	**Yes.**				
MSI (Microsoft Win. Installer)	**Yes**		**Yes**		
MDAC 2.5 or higher	**Yes** -- Load unless IE 5.0 or higher is loaded*				
XML 3.0	**Yes** -- Load unless IE 6.0 (ships with Win XP) or higher is loaded *				

* Note – Even if you already have MDAC 2.5 and XML 3.0, it's a good idea to update to the latest versions if you don't have them. See below.

B. **Steps** to set up EPPeople in a Windows 95 and NT environment will be:

With Windows 95 or NT, loading all of these may require rebooting the computer 3 or 4 times or so. Also, note:

- If the EPPeople setup tells you that you have a more recent component and that you should probably keep it, go ahead and click Yes to keep the newer version on your system.
- If EPPeople setup tells you it needs to update your Windows and reboot, remember that after you do that, you need to run EPPeople setup again.

1. Take a clean machine to test with that already has Word and WinDraft set up and working right.

2. Load any needed Microsoft drivers to bring the Windows components up to current levels.

   a. **DCOM and MSI for Windows 95**. Only needed if you have Windows 95 and they have not been installed by some other program:

      - **DCOM** (for Windows 95) .
        Download from Microsoft (http://www.microsoft.com/com/resources/downloads.asp) or from Lawtech: dcom95.exe. There is also an update for Windows 98.

      - **Microsoft Windows Installer (MSI)** (any version is OK for our needs)

        - New version 2.0 for Windows NT and 2000 (ships with Windows XP): http://www.microsoft.com/downloads/release.asp?releaseid=32832&NewList=1.
        - Version 1.2 for Win 9X and ME: http://www.msdn.microsoft.com/downloads/sample.asp?url=/msdn-files/027/001/455/msdncompositedoc.xml

   b. **MDAC 2.5 or higher**.
   (Included with IE 5.0 and higher. SQL Server 2000 client setup installs 2.6.)
   (See http://support.microsoft.com/view/dev.asp?ID=hl&pg=mdac.asp) for a (FAQ about MDAC, and http://support.microsoft.com/view/dev.asp?ID=hl&pg=mdac.asp to get to the MDAC download page. If you want to go to 2.6 without loading 2.5 first, talk to Jim about the JET issue before you load it or load 2.5, then upgrade to 2.6.)

      Only needed if you don't have IE 5.0 or higher or another program that has loaded current MDAC drivers. The MS Web page has a component checker utility that checks your system for what you have.

   c. **XML 3.0 Parser** (MSXML 3.0)
   Included in IE 6.0 and higher (ships with Win XP).
   As of November 1, 2001, the current version is SP2.
   http://download.microsoft.com/download/xml/SP/3.20/W9X2KMeXP/EN-US/msxml3sp2Setup.exe, or see www.microsoft.com/xml.

3. Run the EPPeople setup.exe (after unzipping it). (Current version 2.0.2).
   Files are on http://eahome.lawtech.com:85/~clifiles/eppeople2/.
   Install EPPeople.exe to the same folder as you have the other WinDraft executables. (Generally c:\program files\windraft).

4. **PRACTICE SYSTEM SETUP.** Run the practice system setup per EA instructions. *(Note: Loading EPPeople without the practice system setup is not enough. The practice system setup is required to create needed ini file settings and copy the necessary files to the system.)*

   The setup file will copy the practice system files to the network or c:\windraft\epxxx (where the epxxx will be your firm's practice system name), and add EPPeople entries into the WinDraft.ini file. See Technical Notes below for discussion of changes to WinDraft.ini.)

5. **TESTING**: Test and troubleshoot any issues.

(Note: When you first open the checklist program, but without any data loaded, the EPPeople program is loaded but hidden. You have to open a data file or click one of the buttons on the People screen in the checklist before the EPPeople program displays.)

- Load EPPeople by itself.
  - From Windows, not WinDraft's checklist, load eppeople. (Start|Programs|WinDraft|EPpeople)
  - Create a set of roles – File|New, Hampton, 0001, and you should get a populated tree.
  - Test as appropriate (drag and drop some roles, then File|Save),
  - Then File|Exit. *(Note: If it works without error, then that means that EPPeople is loaded and working properly, including XML parser and MDAC.)*

- Use a sample data file.

  - Open a sample data file (Hampton)
  - Make sure there are people in the roles table and no error messages (other than possibly the "switch to or retry")
  - Blast off the financial durable power of attorney (FDPOA).

    (Note: If this works, then the OLE Automation aspects are working properly, so that WinDraft's checklist program, WDDATA, can correctly "talk to" EPPeople, and drafting engine, WinDraft.wll, and the forms are correctly reading the EPPeople output.)

- Go through the steps in the "Quick Start to using EPPeople" to create a new one, and blast off. If that works, then you are in business. *(Note: There have been some settings, all involving machines created through a cloning process, such as "Ghost," in which everything worked fine with an existing data file, but creating a new data file caused an error message.)*

C. EXPLANATION.

What's going on with this program is that WDDATA.EXE, written in

C++, and EPPeople.exe, written in VB, talk to each other using Microsoft's Component Object Model (COM) architecture, including database integration using ADO (which requires that current MDAC drivers), and stores the data in XML format. If the DLL's aren't all current and the registry settings aren't all correct, then either the EPPeople program doesn't work right at all, or the integration between the two programs doesn't work well.

It's pretty easy with Windows 2000, Office 2000, and IE 5.5 already installed on a PC, because we just need to install our program and the XML parser (one click, no reboot) and everything works right.

With Win95 and NT 4, it can be tricky, and loading all the components needed to bring Windows up to current DLL's can be a pain. If you can use Ghost or other cloning method to make that work on other machines it really helps, so that you don't have to keep rebooting for each machine.

We should go through this process on a test machine, get it working figure out what adjustments are needed, then probably nuke it and do it again, and then clone that one.

Because of all the rebooting, it's good to multi-task, so that you can be doing something else while waiting for the computer to reboot. Don't waste a morning watching the screen while you load all this stuff.

D. Technical Notes – WinDraft.ini settings

Changes to WinDRaft.ini for EPPeople. The lines indicated in yellow and italics below are the ones that are inserted in the WinDraft.ini file.

The ones in the [WinDraft section are those that store the output options, and are not needed for the program to run properly. (They are created under View|Options.)]

The ones for the practice system are required to support the database access. The path will differ depending on your setup. (network or local, practice system directory name, etc.)

[WinDraft]
ProgDir=C:\Program Files\WinDraft
WinDraftDir=C:\WinDraft
WatchAssembly=1
Author=1
Editor=1
DOCSOpen=0
UserID=

LogFile=C:\WINDOWS\TEMP\wdlog.txt
AlwaysWatch=0
SaveInDMSOn=0
epOptFirstNames=0
epOptUpperCase=1
epOptResidence=0
epOptRelationShips=0
epOptCommaAnd=1
epOptFinalComma=0

[EPXXX-System]
DocDir=C:\WinDraft\EPXXX
SubDir=C:\WinDraft\EPXXX\SUBDOCS
WddDir=C:\WinDraft\EPXXX\DATA

XI. *BMacro=BlastOffEstatePlanningOutputHusbWifeFiles*

BFile=EPXXX.TXT
Script=EPXXX.int
DefTemplate=C:\WinDraft\EPXXX\TEMPLATE\WDFMT.DOT
AltTemplate=C:\WinDraft\EPXXX\TEMPLATE\WDFMTALT.DOT
AltFormat=0
ResetToStyles=0
ChecklistHelp=C:\WinDraft\EPXXX\TEMPLATE\EPXXX.nfo
ScriptHelp=EPXXX.nfo
ChecklistHelpURL=C:\WinDraft\EPXXX\htmlhelp
ScriptHelpURL=C:\WinDraft\EPXXX\htmlhelp
ChecklistTemplate=C:\WinDraft\EPXXX\TEMPLATE\EPXXX.wdt
EPConnectionString=DRIVER=Microsoft Access Driver (.mdb); DBQ=C:\WinDraft\EPXXX\DATA\EPPeople.MDB*
TemplatesDir=C:\WinDraft\EPXXX\TEMPLATE
DocumentsDir=C:\WinDraft\EPXXX\DATA
FileSection=EPXXX-Documents

A. MDAC Drivers

EPPeople requires MDAC 2.5 or higher. If it's not there, you may get an error message like this when you load a data file:

> *"(430) class does not support automation. Or does not support expected interface. Server busy - Lawtech program is not responding. Click appropriate to activate program and correct problem."*

Eppeople then comes up not populated.

XII. Authoring -- Other Mirror-Image Functions

We added a function to EPPeople.exe that doesn't really have anything to do with people and roles. It makes it much easier for an author to ask separate questions for husband and wife, and have the system automatically plug the right information for the husband or wife when the documents are "blasted off."

Most of the time, other than people and their roles, the same questions and answers are used for the husband and wife's documents.

There are a number of situations in which you want to ask separate questions for the husband and wife, such as specific gifts.

In the Checklist	**In the Model Document**
Precede variable name with ".husb" or ."wife"	Do not use Husb. Or Wife. In the variable name.
Example: Husb.SpecificGifts Wife.SpecificGifts	Example: [SpecificGifts]

This is handled in the Blastoff Macro called EstatePlanningBlastoffOutputHusbWifeFiles. It outputs two files, one for husb and one for wife that can be pulled in with a GETDATA command.

XIII. Appendix – More Authoring Examples

A. Will

ARTICLE VII
APPOINTMENT OF PERSONAL REPRESENTATIVE

I appoint [PR(1).Names] to be my personal representative.

[IF PR(2).count > "0"][;; if there is a 2nd level PR][;; if levels = exactly 2, put it in the para. If more than 2, put them in a list.]
If [PR(1).Names] cannot act, or [isare(PR(1))] unwilling to act, I appoint[IF PR.Levels = "2"][PR(2).Names].
[ELSE
], in the order named, the following successor personal representatives:

[FOR I = 2 TO PR.Levels INCLUDE "PersRep.doc"][;

][ENDIF]
[ENDIF]

I direct that my personal representatives not be required to furnish bond, surety, or other security.

[IF MinorChildYN = "Y"]

ARTICLE VII
APPOINTMENT OF GUARDIAN

*[IF GrantorIsMarried = "Y"]*If my spouse does not survive me, *[ENDIF]*I appoint [Guardian(1).Name] to be the guardian of my minor children.

[IF Guardian.Levels > "1"]

If [HeShe(Guardian(1))] is unable or unwilling to serve, I appoint [Guardian(2).Name] as my successor guardian.

[ENDIF]

[ENDIF][;; minor children]

XIV. Authoring – The EPPeople Output

A. Getdata Commands

Note that each template with have to get the data that is relevant to that document.

Here are guidelines:

- Always get the data for that specific document if there are roles defined in that document. (Eg. Husb.will.txt for the Husband's Will template, WillHusb.dot).
- If you are doing the husband's documents (or joint), get Husb.txt. If you are doing the wife's document (not joint), get Wife.txt.
- If you are using data relating to the other spouse or otherwise need or you need Husb. And Wife. info, get WDOutputAll.txt.

B. Examples

1. The User's View:

2. The Blastoff File temporary Data

At blastoff, the system will output a number of temporary "blast off data files" that WinDraft reads using a GETDATA command.

There are 3 levels of files that contain the same data with different "paths" for the variables.

WDOutputAll.txt
Husb.txt and Wife.txt
Husb.Will.Txt and other document-specific information

Here is the ouput with the full path for each variable for the agents in the screen shown above.

a) In WDOutputAll.txt

husb.FDPOA.Agent(1).Count = 2
husb.FDPOA.Agent(1).Names = JILL LYNN HAMPTON and ROBERT JOHNSON
husb.FDPOA.Agent(1).Name = JILL LYNN HAMPTON and ROBERT JOHNSON
husb.FDPOA.Agent(1).Person(1).RecordID = 172
husb.FDPOA.Agent(1).Person(1).LastName = Hampton
husb.FDPOA.Agent(1).Person(1).FirstName = Jill
husb.FDPOA.Agent(1).Person(1).FullName = Jill Lynn Hampton
husb.FDPOA.Agent(1).Person(1).Middle = Lynn
husb.FDPOA.Agent(1).Person(1).RelToHusb = wife
husb.FDPOA.Agent(1).Person(1).RelToWife = self
husb.FDPOA.Agent(1).Person(1).Gender = F
husb.FDPOA.Agent(1).Person(1).ClientMatter = UNANSWERED
husb.FDPOA.Agent(1).Person(1).ClientID = Hampton
husb.FDPOA.Agent(1).Person(1).MatterID = 0001

husb.FDPOA.Agent(1).Person(1).DOBMMDDYYYY = UNANSWERED
husb.FDPOA.Agent(1).Person(1).DateOfBirth = UNANSWERED
husb.FDPOA.Agent(1).Person(1).DateOfDeath = UNANSWERED
husb.FDPOA.Agent(1).Person(1).Street1 = 4329 Timberdale Road
husb.FDPOA.Agent(1).Person(1).Street2 = UNANSWERED
husb.FDPOA.Agent(1).Person(1).City = Miami
husb.FDPOA.Agent(1).Person(1).ST = FL
husb.FDPOA.Agent(1).Person(1).State = Florida
husb.FDPOA.Agent(1).Person(1).Zip = 55555
husb.FDPOA.Agent(1).Person(1).Code = UNANSWERED
husb.FDPOA.Agent(1).Person(1).BusPhone = UNANSWERED
husb.FDPOA.Agent(1).Person(1).HomePhone = (305) 532-1457
husb.FDPOA.Agent(1).Person(1).EMail = UNANSWERED
husb.FDPOA.Agent(1).Person(1).Pager = UNANSWERED
husb.FDPOA.Agent(1).Person(1).Fax = UNANSWERED
husb.FDPOA.Agent(1).Person(1).MaritalStatus = UNANSWERED
husb.FDPOA.Agent(1).Person(1).AKAName = UNANSWERED
husb.FDPOA.Agent(1).Person(1).NickName = UNANSWERED
husb.FDPOA.Agent(1).Person(1).Prefix = UNANSWERED
husb.FDPOA.Agent(1).Person(1).Salutation = UNANSWERED
husb.FDPOA.Agent(1).Person(1).SalutationCouple = UNANSWERED
husb.FDPOA.Agent(1).Person(1).Suffix = UNANSWERED
husb.FDPOA.Agent(1).Person(1).CityOrCounty = UNANSWERED
husb.FDPOA.Agent(1).Person(1).County = Miami-Dade
husb.FDPOA.Agent(1).Person(1).Country = UNANSWERED
husb.FDPOA.Agent(1).Person(1).GovernmentID = UNANSWERED
husb.FDPOA.Agent(1).Person(1).Citizen = Cuba
husb.FDPOA.Agent(1).Person(1).CompetencyCode = UNANSWERED
husb.FDPOA.Agent(1).Person(1).AgentName = UNANSWERED
husb.FDPOA.Agent(1).Person(1).LivingArrangements = UNANSWERED
husb.FDPOA.Agent(1).Person(1).ClientOfFirm = UNANSWERED
husb.FDPOA.Agent(1).Person(1).PersonNotes = UNANSWERED
husb.FDPOA.Agent(1).Person(1).Name = JILL LYNN HAMPTON
husb.FDPOA.Agent(1).Person(1).Index = 2
husb.FDPOA.Agent(1).Person(1) = 2
husb.FDPOA.Agent(1).Person(1).Relationship = wife
husb.FDPOA.Agent(1).Person(2).RecordID = 167
husb.FDPOA.Agent(1).Person(2).LastName = Johnson
husb.FDPOA.Agent(1).Person(2).FirstName = Robert
husb.FDPOA.Agent(1).Person(2).FullName = Robert Johnson
husb.FDPOA.Agent(1).Person(2).Middle = UNANSWERED
husb.FDPOA.Agent(1).Person(2).RelToHusb = friend
husb.FDPOA.Agent(1).Person(2).RelToWife = friend
husb.FDPOA.Agent(1).Person(2).Gender = M
husb.FDPOA.Agent(1).Person(2).ClientMatter = UNANSWERED
husb.FDPOA.Agent(1).Person(2).ClientID = Hampton

husb.FDPOA.Agent(1).Person(2).MatterID = 0001
husb.FDPOA.Agent(1).Person(2).DOBMMDDYYYY = UNANSWERED
husb.FDPOA.Agent(1).Person(2).DateOfBirth = UNANSWERED
husb.FDPOA.Agent(1).Person(2).DateOfDeath = UNANSWERED
husb.FDPOA.Agent(1).Person(2).Street1 = 6969 Atom Avenue
husb.FDPOA.Agent(1).Person(2).Street2 = UNANSWERED
husb.FDPOA.Agent(1).Person(2).City = Studio City
husb.FDPOA.Agent(1).Person(2).ST = CA
husb.FDPOA.Agent(1).Person(2).State = California
husb.FDPOA.Agent(1).Person(2).Zip = 90069
husb.FDPOA.Agent(1).Person(2).Code = UNANSWERED
husb.FDPOA.Agent(1).Person(2).BusPhone = UNANSWERED
husb.FDPOA.Agent(1).Person(2).HomePhone = (805) 234-9870
husb.FDPOA.Agent(1).Person(2).EMail = UNANSWERED
husb.FDPOA.Agent(1).Person(2).Pager = UNANSWERED
husb.FDPOA.Agent(1).Person(2).Fax = UNANSWERED
husb.FDPOA.Agent(1).Person(2).MaritalStatus = UNANSWERED
husb.FDPOA.Agent(1).Person(2).AKAName = UNANSWERED
husb.FDPOA.Agent(1).Person(2).NickName = UNANSWERED
husb.FDPOA.Agent(1).Person(2).Prefix = UNANSWERED
husb.FDPOA.Agent(1).Person(2).Salutation = UNANSWERED
husb.FDPOA.Agent(1).Person(2).SalutationCouple = UNANSWERED
husb.FDPOA.Agent(1).Person(2).Suffix = UNANSWERED
husb.FDPOA.Agent(1).Person(2).CityOrCounty = UNANSWERED
husb.FDPOA.Agent(1).Person(2).County = Los Angeles
husb.FDPOA.Agent(1).Person(2).Country = UNANSWERED
husb.FDPOA.Agent(1).Person(2).GovernmentID = UNANSWERED
husb.FDPOA.Agent(1).Person(2).Citizen = UNANSWERED
husb.FDPOA.Agent(1).Person(2).CompetencyCode = UNANSWERED
husb.FDPOA.Agent(1).Person(2).AgentName = UNANSWERED
husb.FDPOA.Agent(1).Person(2).LivingArrangements = UNANSWERED
husb.FDPOA.Agent(1).Person(2).ClientOfFirm = UNANSWERED
husb.FDPOA.Agent(1).Person(2).PersonNotes = UNANSWERED
husb.FDPOA.Agent(1).Person(2).Name = ROBERT JOHNSON
husb.FDPOA.Agent(1).Person(2).Index = 1
husb.FDPOA.Agent(1).Person(2) = 1
husb.FDPOA.Agent(1).Person(2).Relationship = friend
husb.FDPOA.Agent(1).Index = 3
husb.FDPOA.Agent(1) = 3
husb.FDPOA.Agent(1).TypeCode = SO
husb.FDPOA.Agent(2).Count = 1
husb.FDPOA.Agent(2).Names = JEFFREY M. SMITH
husb.FDPOA.Agent(2).Name = JEFFREY M. SMITH
husb.FDPOA.Agent(2).Person(1).RecordID = 173
husb.FDPOA.Agent(2).Person(1).LastName = Smith
husb.FDPOA.Agent(2).Person(1).FirstName = Jeffrey

husb.FDPOA.Agent(2).Person(1).FullName = Jeffrey M. Smith
husb.FDPOA.Agent(2).Person(1).Middle = M.
husb.FDPOA.Agent(2).Person(1).RelToHusb = UNANSWERED
husb.FDPOA.Agent(2).Person(1).RelToWife = UNANSWERED
husb.FDPOA.Agent(2).Person(1).Gender = M
husb.FDPOA.Agent(2).Person(1).ClientMatter = UNANSWERED
husb.FDPOA.Agent(2).Person(1).ClientID = Common
husb.FDPOA.Agent(2).Person(1).MatterID = EP
husb.FDPOA.Agent(2).Person(1).DOBMMDDYYYY = UNANSWERED
husb.FDPOA.Agent(2).Person(1).DateOfBirth = UNANSWERED
husb.FDPOA.Agent(2).Person(1).DateOfDeath = UNANSWERED
husb.FDPOA.Agent(2).Person(1).Street1 = 4510 E. Thousand Oaks Boulevard
husb.FDPOA.Agent(2).Person(1).Street2 = Suite 200
husb.FDPOA.Agent(2).Person(1).City = Westlake Village
husb.FDPOA.Agent(2).Person(1).ST = CA
husb.FDPOA.Agent(2).Person(1).State = California
husb.FDPOA.Agent(2).Person(1).Zip = 55555
husb.FDPOA.Agent(2).Person(1).Code = UNANSWERED
husb.FDPOA.Agent(2).Person(1).BusPhone = UNANSWERED
husb.FDPOA.Agent(2).Person(1).HomePhone = UNANSWERED
husb.FDPOA.Agent(2).Person(1).EMail = UNANSWERED
husb.FDPOA.Agent(2).Person(1).Pager = UNANSWERED
husb.FDPOA.Agent(2).Person(1).Fax = UNANSWERED
husb.FDPOA.Agent(2).Person(1).MaritalStatus = UNANSWERED
husb.FDPOA.Agent(2).Person(1).AKAName = UNANSWERED
husb.FDPOA.Agent(2).Person(1).NickName = UNANSWERED
husb.FDPOA.Agent(2).Person(1).Prefix = UNANSWERED
husb.FDPOA.Agent(2).Person(1).Salutation = UNANSWERED
husb.FDPOA.Agent(2).Person(1).SalutationCouple = UNANSWERED
husb.FDPOA.Agent(2).Person(1).Suffix = UNANSWERED
husb.FDPOA.Agent(2).Person(1).CityOrCounty = UNANSWERED
husb.FDPOA.Agent(2).Person(1).County = Ventura
husb.FDPOA.Agent(2).Person(1).Country = UNANSWERED
husb.FDPOA.Agent(2).Person(1).GovernmentID = UNANSWERED
husb.FDPOA.Agent(2).Person(1).Citizen = UNANSWERED
husb.FDPOA.Agent(2).Person(1).CompetencyCode = UNANSWERED
husb.FDPOA.Agent(2).Person(1).AgentName = UNANSWERED
husb.FDPOA.Agent(2).Person(1).LivingArrangements = UNANSWERED
husb.FDPOA.Agent(2).Person(1).ClientOfFirm = UNANSWERED
husb.FDPOA.Agent(2).Person(1).PersonNotes = UNANSWERED
husb.FDPOA.Agent(2).Person(1).Name = JEFFREY M. SMITH
husb.FDPOA.Agent(2).Person(1).Index = 1
husb.FDPOA.Agent(2).Person(1) = 1
husb.FDPOA.Agent(2).Person(1).Relationship =
husb.FDPOA.Agent(2).Index = 1
husb.FDPOA.Agent(2) = 1

husb.FDPOA.Agent(2).Person(2).Index = 0
husb.FDPOA.Agent(2).Person(2).Name = UNANSWERED
husb.FDPOA.Agent(2).TypeCode = O
husb.FDPOA.Agent.Levels = 2
husb.FDPOA.Agent.Count = 2

C. In Husb.FDPOA.txt

Agent(1).Count = 2
Agent(1).Names = JILL LYNN HAMPTON, and ROBERT JOHNSON
Agent(1).Name = JILL LYNN HAMPTON, and ROBERT JOHNSON
Agent(1).Person(1).RecordID = 172
Agent(1).Person(1).LastName = Hampton
Agent(1).Person(1).FirstName = Jill
Agent(1).Person(1).FullName = Jill Lynn Hampton
Agent(1).Person(1).Middle = Lynn
Agent(1).Person(1).RelToHusb = wife
Agent(1).Person(1).RelToWife = self
Agent(1).Person(1).Gender = F
Agent(1).Person(1).ClientMatter = UNANSWERED
Agent(1).Person(1).ClientID = Hampton
Agent(1).Person(1).MatterID = 0001
Agent(1).Person(1).DOBMMDDYYYY = 12/17/46
Agent(1).Person(1).DateOfBirth = December 17, 1946
Agent(1).Person(1).DateOfDeath = UNANSWERED
Agent(1).Person(1).Street1 = 4329 Timberdale Road
Agent(1).Person(1).Street2 = UNANSWERED
Agent(1).Person(1).City = Miami
Agent(1).Person(1).ST = FL
Agent(1).Person(1).State = Florida
Agent(1).Person(1).Zip = 55555
Agent(1).Person(1).Code = UNANSWERED
Agent(1).Person(1).BusPhone = UNANSWERED
Agent(1).Person(1).HomePhone = (305) 532-1457
Agent(1).Person(1).EMail = UNANSWERED
Agent(1).Person(1).Pager = UNANSWERED
Agent(1).Person(1).Fax = UNANSWERED
Agent(1).Person(1).MaritalStatus = UNANSWERED
Agent(1).Person(1).AKAName = UNANSWERED
Agent(1).Person(1).NickName = UNANSWERED
Agent(1).Person(1).Prefix = UNANSWERED
Agent(1).Person(1).Salutation = UNANSWERED
Agent(1).Person(1).SalutationCouple = UNANSWERED

Agent(1).Person(1).Suffix = UNANSWERED
Agent(1).Person(1).CityOrCounty = UNANSWERED
Agent(1).Person(1).County = Miami-Dade
Agent(1).Person(1).Country = UNANSWERED
Agent(1).Person(1).GovernmentID = UNANSWERED
Agent(1).Person(1).Citizen = Cuba
Agent(1).Person(1).CompetencyCode = UNANSWERED
Agent(1).Person(1).AgentName = UNANSWERED
Agent(1).Person(1).LivingArrangements = UNANSWERED
Agent(1).Person(1).ClientOfFirm = UNANSWERED
Agent(1).Person(1).PersonNotes = UNANSWERED
Agent(1).Person(1).Name = JILL LYNN HAMPTON
Agent(1).Person(1).Index = 2
Agent(1).Person(1) = 2
Agent(1).Person(1).Relationship = wife
Agent(1).Person(2).RecordID = 167
Agent(1).Person(2).LastName = Johnson
Agent(1).Person(2).FirstName = Robert
Agent(1).Person(2).FullName = Robert Johnson
Agent(1).Person(2).Middle = UNANSWERED
Agent(1).Person(2).RelToHusb = friend
Agent(1).Person(2).RelToWife = friend
Agent(1).Person(2).Gender = M
Agent(1).Person(2).ClientMatter = UNANSWERED
Agent(1).Person(2).ClientID = Hampton
Agent(1).Person(2).MatterID = 0001
Agent(1).Person(2).DOBMMDDYYYY = UNANSWERED
Agent(1).Person(2).DateOfBirth = UNANSWERED
Agent(1).Person(2).DateOfDeath = UNANSWERED
Agent(1).Person(2).Street1 = 6969 Atom Avenue
Agent(1).Person(2).Street2 = UNANSWERED
Agent(1).Person(2).City = Studio City
Agent(1).Person(2).ST = CA
Agent(1).Person(2).State = California
Agent(1).Person(2).Zip = 90069
Agent(1).Person(2).Code = UNANSWERED
Agent(1).Person(2).BusPhone = UNANSWERED
Agent(1).Person(2).HomePhone = (805) 234-9870
Agent(1).Person(2).EMail = UNANSWERED
Agent(1).Person(2).Pager = UNANSWERED
Agent(1).Person(2).Fax = UNANSWERED
Agent(1).Person(2).MaritalStatus = UNANSWERED
Agent(1).Person(2).AKAName = UNANSWERED
Agent(1).Person(2).NickName = UNANSWERED
Agent(1).Person(2).Prefix = UNANSWERED
Agent(1).Person(2).Salutation = UNANSWERED

Agent(1).Person(2).SalutationCouple = UNANSWERED
Agent(1).Person(2).Suffix = UNANSWERED
Agent(1).Person(2).CityOrCounty = UNANSWERED
Agent(1).Person(2).County = Los Angeles
Agent(1).Person(2).Country = UNANSWERED
Agent(1).Person(2).GovernmentID = UNANSWERED
Agent(1).Person(2).Citizen = UNANSWERED
Agent(1).Person(2).CompetencyCode = UNANSWERED
Agent(1).Person(2).AgentName = UNANSWERED
Agent(1).Person(2).LivingArrangements = UNANSWERED
Agent(1).Person(2).ClientOfFirm = UNANSWERED
Agent(1).Person(2).PersonNotes = UNANSWERED
Agent(1).Person(2).Name = ROBERT JOHNSON
Agent(1).Person(2).Index = 1
Agent(1).Person(2) = 1
Agent(1).Person(2).Relationship = friend
Agent(1).Index = 3
Agent(1) = 3
Agent(1).TypeCode = SO
Agent(2).Count = 1
Agent(2).Names = JEFFREY M. SMITH
Agent(2).Name = JEFFREY M. SMITH
Agent(2).Person(1).RecordID = 173
Agent(2).Person(1).LastName = Smith
Agent(2).Person(1).FirstName = Jeffrey
Agent(2).Person(1).FullName = Jeffrey M. Smith
Agent(2).Person(1).Middle = M.
Agent(2).Person(1).RelToHusb = UNANSWERED
Agent(2).Person(1).RelToWife = UNANSWERED
Agent(2).Person(1).Gender = M
Agent(2).Person(1).ClientMatter = UNANSWERED
Agent(2).Person(1).ClientID = Common
Agent(2).Person(1).MatterID = EP
Agent(2).Person(1).DOBMMDDYYYY = UNANSWERED
Agent(2).Person(1).DateOfBirth = UNANSWERED
Agent(2).Person(1).DateOfDeath = UNANSWERED
Agent(2).Person(1).Street1 = 4510 E. Thousand Oaks Boulevard
Agent(2).Person(1).Street2 = Suite 200
Agent(2).Person(1).City = Westlake Village
Agent(2).Person(1).ST = CA
Agent(2).Person(1).State = California
Agent(2).Person(1).Zip = 55555
Agent(2).Person(1).Code = UNANSWERED
Agent(2).Person(1).BusPhone = UNANSWERED
Agent(2).Person(1).HomePhone = UNANSWERED
Agent(2).Person(1).EMail = UNANSWERED

Agent(2).Person(1).Pager = UNANSWERED
Agent(2).Person(1).Fax = UNANSWERED
Agent(2).Person(1).MaritalStatus = UNANSWERED
Agent(2).Person(1).AKAName = UNANSWERED
Agent(2).Person(1).NickName = UNANSWERED
Agent(2).Person(1).Prefix = UNANSWERED
Agent(2).Person(1).Salutation = UNANSWERED
Agent(2).Person(1).SalutationCouple = UNANSWERED
Agent(2).Person(1).Suffix = UNANSWERED
Agent(2).Person(1).CityOrCounty = UNANSWERED
Agent(2).Person(1).County = Ventura
Agent(2).Person(1).Country = UNANSWERED
Agent(2).Person(1).GovernmentID = UNANSWERED
Agent(2).Person(1).Citizen = UNANSWERED
Agent(2).Person(1).CompetencyCode = UNANSWERED
Agent(2).Person(1).AgentName = UNANSWERED
Agent(2).Person(1).LivingArrangements = UNANSWERED
Agent(2).Person(1).ClientOfFirm = UNANSWERED
Agent(2).Person(1).PersonNotes = UNANSWERED
Agent(2).Person(1).Name = JEFFREY M. SMITH
Agent(2).Person(1).Index = 1
Agent(2).Person(1) = 1
Agent(2).Person(1).Relationship =
Agent(2).Index = 1
Agent(2) = 1
Agent(2).Person(2).Index = 0
Agent(2).Person(2).Name = UNANSWERED
Agent(2).TypeCode = O
Agent.Levels = 2
Agent.Count = 2

QUICK START TO USING WDDATA AND EPPEOPLE TOGETHER.

by JAE -- July 3, 2001

I. INTRODUCTION

A. BACKGROUND

WDDATA.EXE is the "WinDraft Data Entry" Program, which we also refer to as the WinDraft "Checklist." See *Quick Start to Using WinDraft* and the *WinDraft User's Guide* for more information on that.

This guide adds "quick start" information for lawyers, legal assistants, and legal secretaries on how to use the *EPPeople* "fiduciary chooser" program, which works in conjunction with the other WinDraft components.

See "*EPPeople Fiduciary Chooser User Manual* for complete information, including a full description of what this module does and how to use it, how to program it, and how to setup and support it. If you haven't done so, we recommend reading at least the beginning of that document before you read this so that you can see what the program does and how it helps you.

B. INTEGRATION WITH THE WINDRAFT CHECKLIST

Although EPPeople can be used independently, this documentation assumes that you will be using EPPeople with full WinDraft checklist integration. Thus, you perform most file operations for EPPeople within WDDATA.

C. SCREEN SHOTS MAY DIFFER

Note that your screens and systems may vary slightly from what you see in this document. If so, that is because either your practice system's integration is different, or because we have added features to EPPeople and WinDraft. Either way, we hope that if you follow the steps described here, you will be able to work with the programs as installed in your firm.

II. WORKING WITH DATA

First of all, this documentation assumes that you know how to use WinDraft, independent of EPPeople. **If you don't, please read the Quick Start to Using WinDraft.**

A. WORKING WITH AN EXISTING DATA FILE.

When you open the WinDraft Data Entry (Checklist) program, the EPPeople program will load automatically, but it will be hidden until you are working with data.

Click on File|Open to open an existing data file, such as the demo data, Hampton1.wdd.

When you do, WDDATA.exe will both open the data file and cause EPPeople.exe to access the "cast of characters" in the database and their roles for that data file.

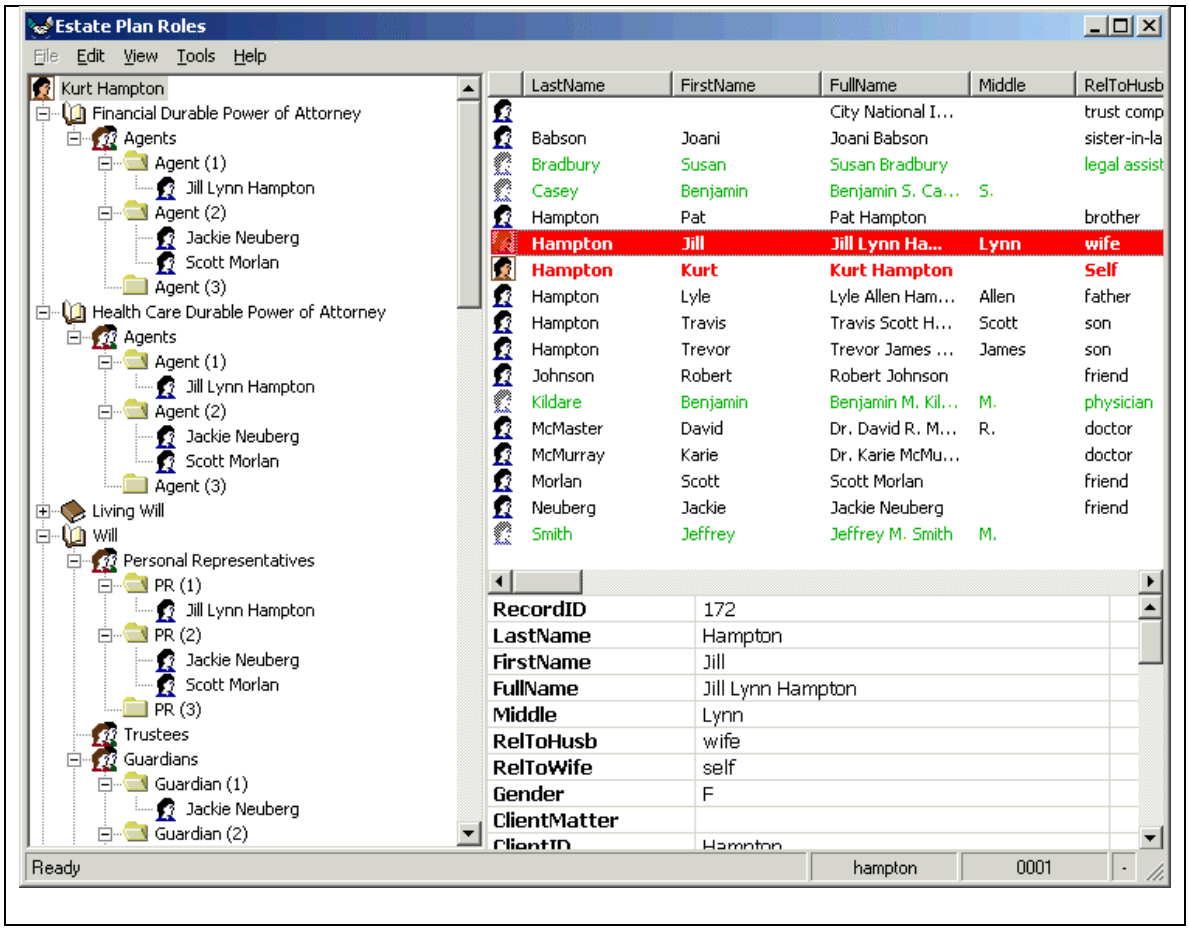
NOTE: Although WDDATA.exe and EPPeople.exe work together, they are two separate programs, and you often need to move back and forth between the two using standard Windows commands for moving between windows.

You can either:

- use the A T keys to switch back and forth between the WinDraft checklist and the EPPeople windows, **or**
- click on the Windows task bar at the bottom of the screen, which looks something like this:

(And note that with Windows NT and 2000, sometimes WinDraft can't switch the "focus" to the EPPeople window, and you need to manually switch windows.)

The EPPeople screen will look something like this:

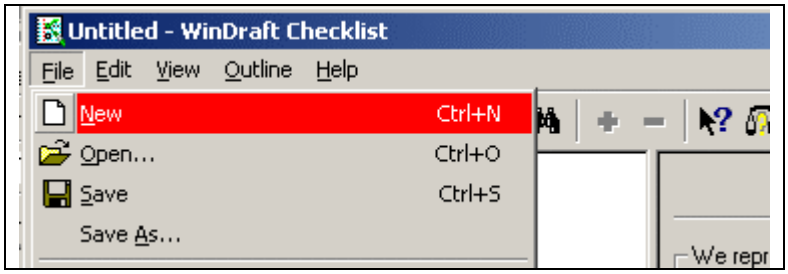


See the full EPPeople user manual for a full description of how to drag and drop data and work with the other options.

When the data is the way you want it to be, switch back to the WinDraft checklist, and click "blastoff." (Use Alt-Tab or click on the WinDraft checklist icon on the Windows task bar at the bottom of the screen to switch between the programs.)

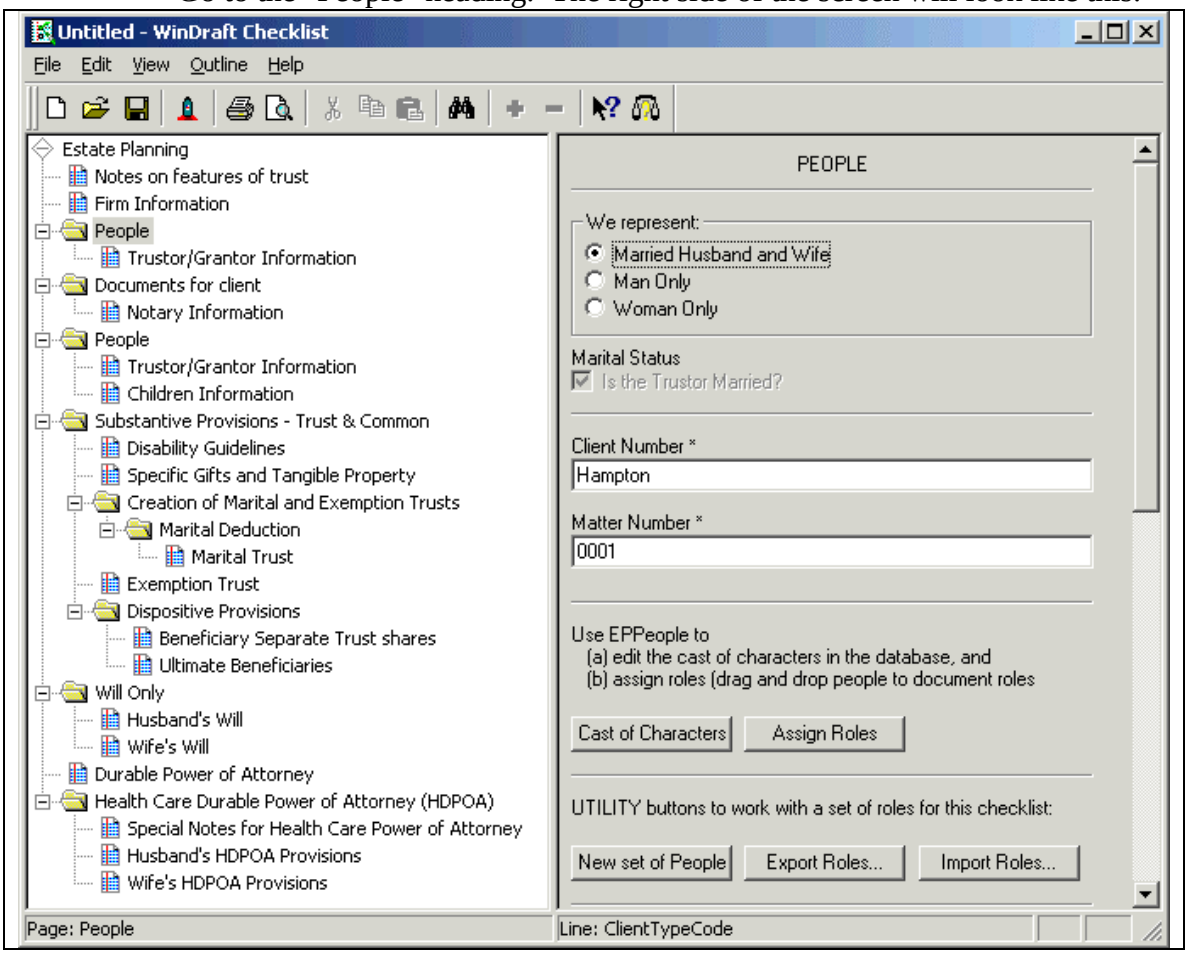
B. WORKING WITH A NEW DATA FILE

1. IN THE WINDRAFT CHECKLIST, CLICK FILE|NEW TO CREATE A NEW DATA FILE.



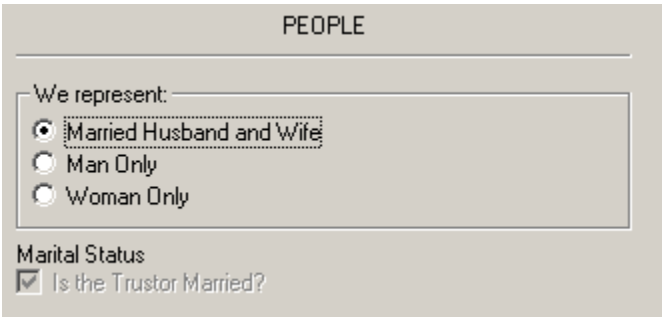
2. CLICK ON THE "PEOPLE" HEADING.

Go to the "People" heading. The right side of the screen will look like this:



(Note that your practice system may have these questions and buttons on different screens, such as FileInfo, depending on how your system has been set up.)

3. ANSWER THE QUESTIONS ABOUT THE MARITAL STATUS OF THE CLIENT:



(If you don't answer these questions when you create your EPPeople roles, the system will assume that the client is married and use the template for married people when it creates the document roles. If you forget to click single, that's OK. You can always delete the unneeded roles later.)

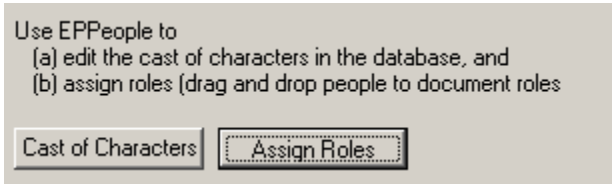
4. ENTER THE CLIENT AND MATTER CODE.

Note that this can be letters or numbers. If you practice in a large firm, it would typically be something like 100234 and 0001. In a smaller firm, you might just insert the client's name as the "Client Number."

Note that once you enter the client and matter code for a group of people, they stick, and you can't change them without a hassle. Therefore, make sure you enter the client number correctly.

5. ENTER PEOPLE IN THE DATABASE

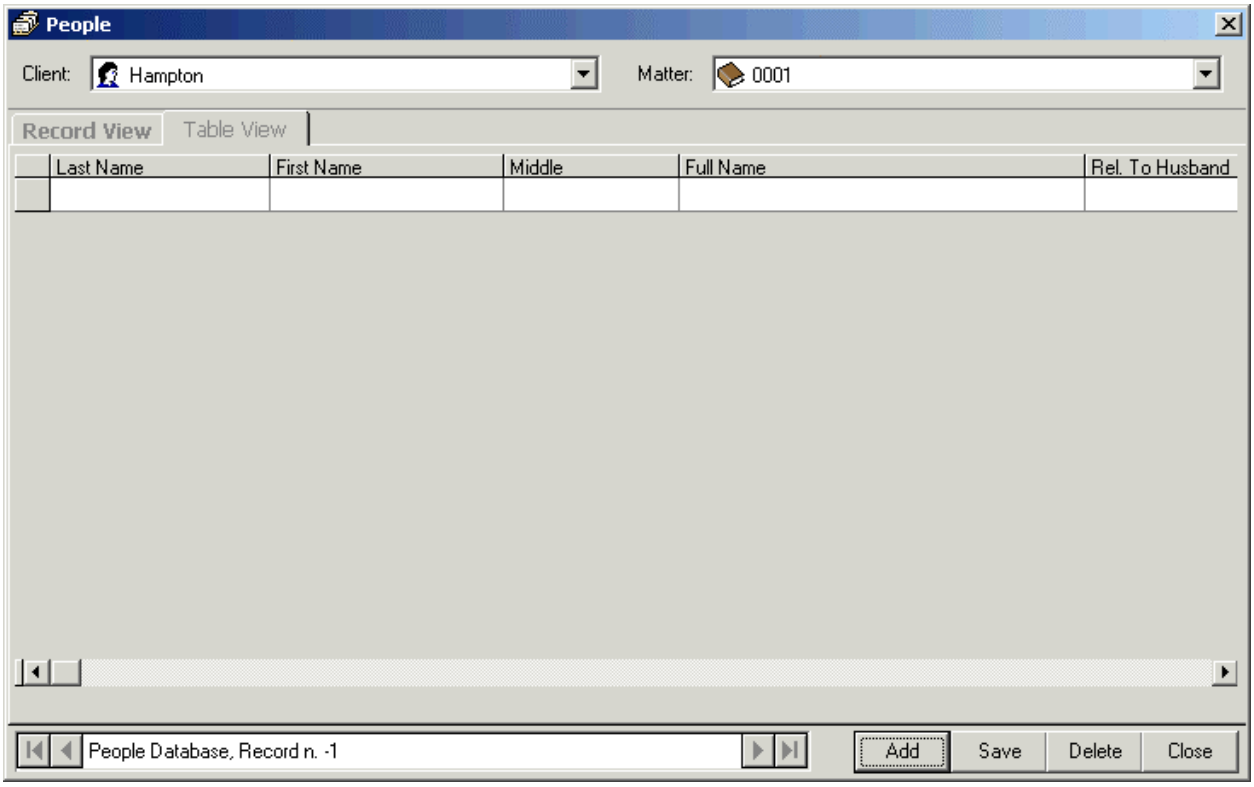
a) Click the Cast of Characters button to load EPPeople


Use EPPeople to
(a) edit the cast of characters in the database, and
(b) assign roles (drag and drop people to document roles
Cast of Characters
Assign Roles

If you click on "Cast of Characters," the system will switch the focus from the WinDraft checklist to the EPPeople program's data entry screen for entering people into the database.

(Note: If you are already in the EPPeople program, or if you click on Assign Roles first, which takes you into the EPPeople program and the "roles tree," you can get to the same screens to enter people into the database by clicking the View|People menu choice.)

The screen to enter people looks like this:



	The icon on the task bar at the bottom of your Windows screen will look like this:

Click the "Add" button at the bottom of the screen to add a person to the database.

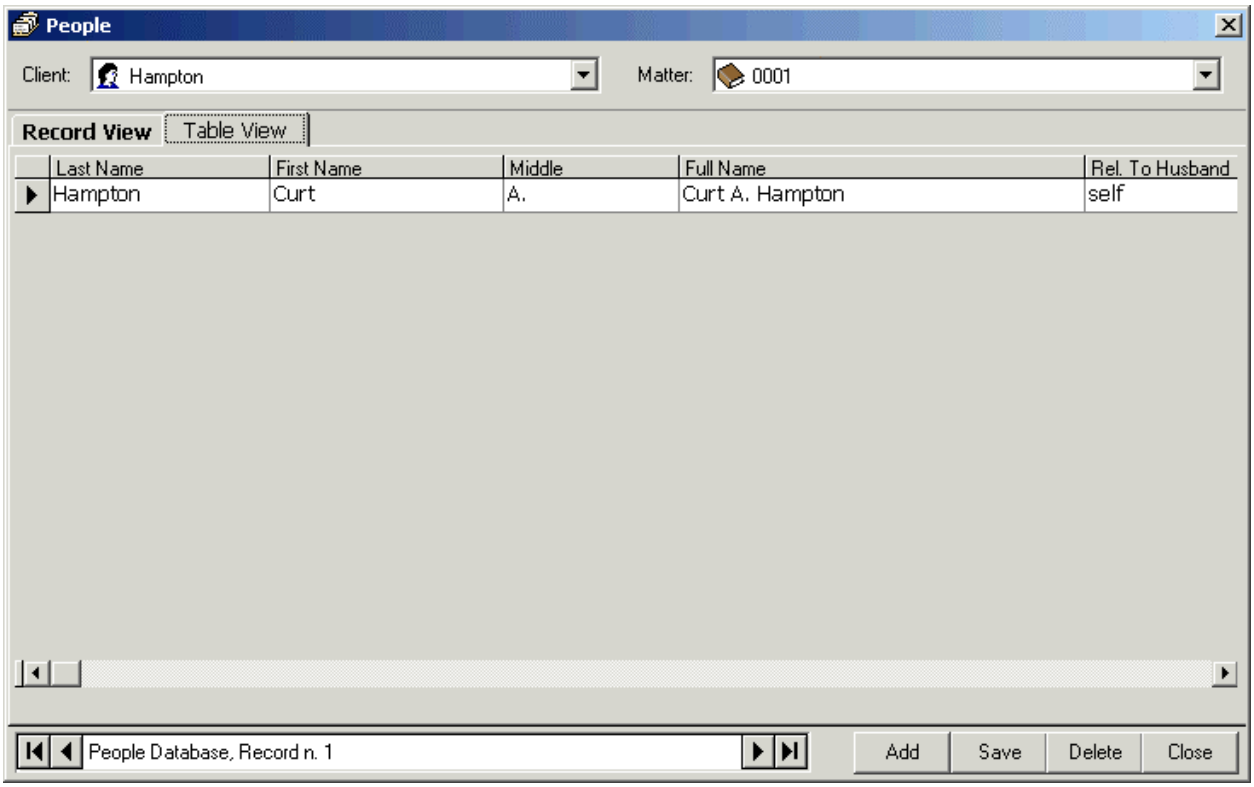
You'll see that the grayed-out tabs for "record view" and "table view" become black. You can click on the tab to move back and forth between record view and table view.

(In either view, the Add, Save, Delete and Close buttons at the bottom of the screen work the same.)

(1) Table View

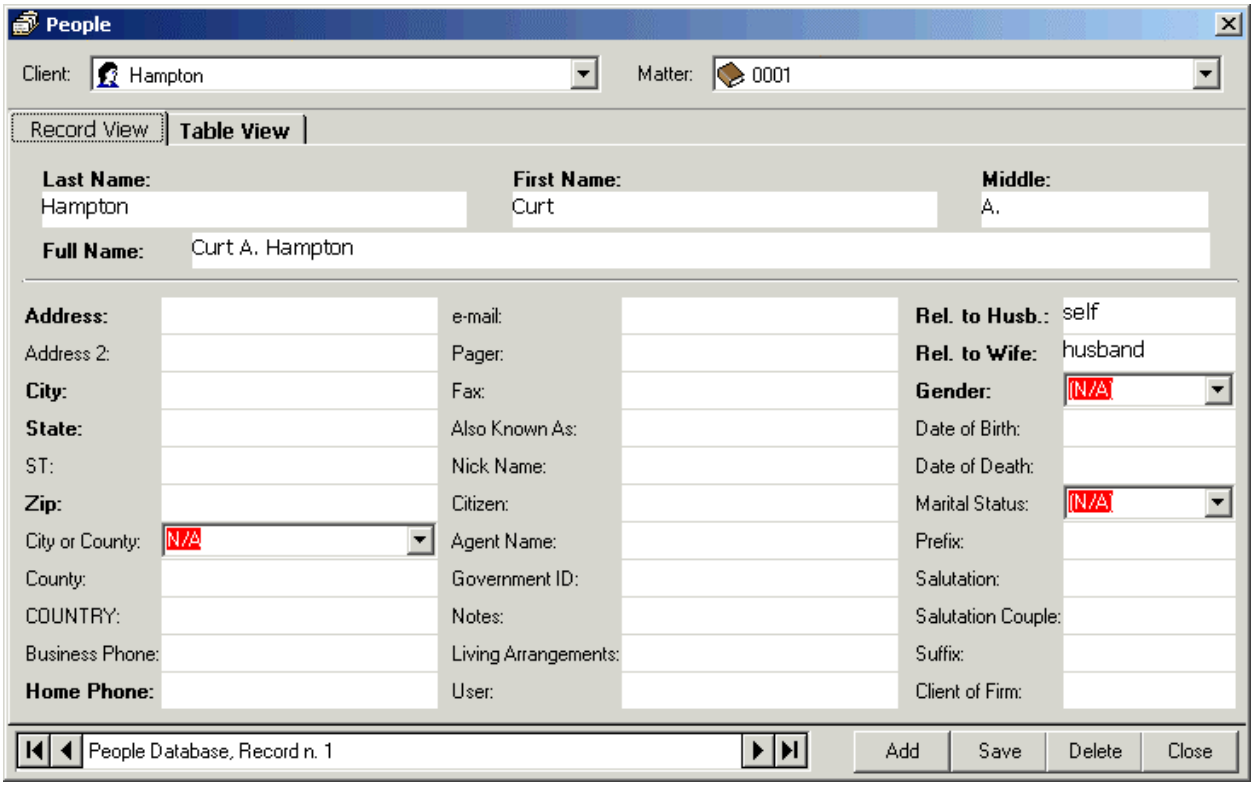
Table view allows you to see all of the people on one screen, but you can't see all of their data unless you scroll to the right.

This is "Table View:" As you add more records, you'll see them in a table that looks like an Excel spreadsheet. You can scroll to the right to add and edit the data. Each time you want to add another record, click the "Add" button.



(2) Record View

Record view lets you see all of the data for just one person at a time, but you can see all of the data for that person, like this:



(3) Navigating through the Records.

If you want to go back and forward through the people, there are two ways to navigate. Either go into table view and scroll up and down through the list. Or, in record view, you can click the arrows at the bottom of the screen.

You can see what record number you are on, and the arrows let you to the beginning, the previous record, the next record, or the end, respectively.

(4) Data You Must Enter

Be sure to enter at least the name information, gender, state, and relationship information. For relationship, enter both relationship to the husband and to the wife. This is critical, and must be entered, at least for the clients and their children.

Clients: For the man, enter "self" for relationship to husband. For the woman, enter "self" for relationship to wife. (You must do this, even if the client is single.)

Children: Use "**son**" or "**daughter**" for the Relations to Husband and Relationship to Wife. The system will also output special data for stepchildren, so if your firm as implemented a system that handles these, you can also use "**step-son**" and "**step-daughter**."

At least for the clients, be sure to enter **state**. (Enter state abbreviation in the ST field, and the state spelled out in the State field.)

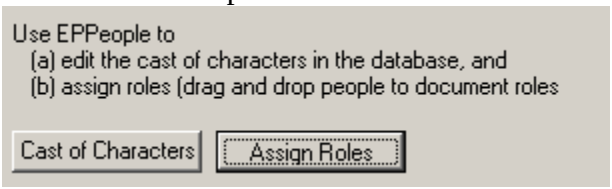
The rest of the fields are optional. (Of course, you should enter any fields that are needed in your practice systems.) If addresses are needed in your firm's documents, be sure to enter them.

See the full EPPeople User Manual for more information on entering these fields and how they are used.

(5) When you are done, click "Close."

6. ASSIGN ROLES USING DRAG AND DROP INTERFACE.

If you are in the WinDraft checklist interface, click on the "Assign Roles" button on the People section of the outline.


Use EPPeople to
(a) edit the cast of characters in the database, and
(b) assign roles (drag and drop people to document roles
Cast of Characters
Assign Roles

If you are in EPPeople already from having entered the people, simply begin dragging and dropping people from the list on the right side to the roles on the left side. (If you can't see the documents and all of their roles, click on them, and they open up. Or click on a node, then click "Edit|Expand Node" on the menu.

a) Drag and Drop

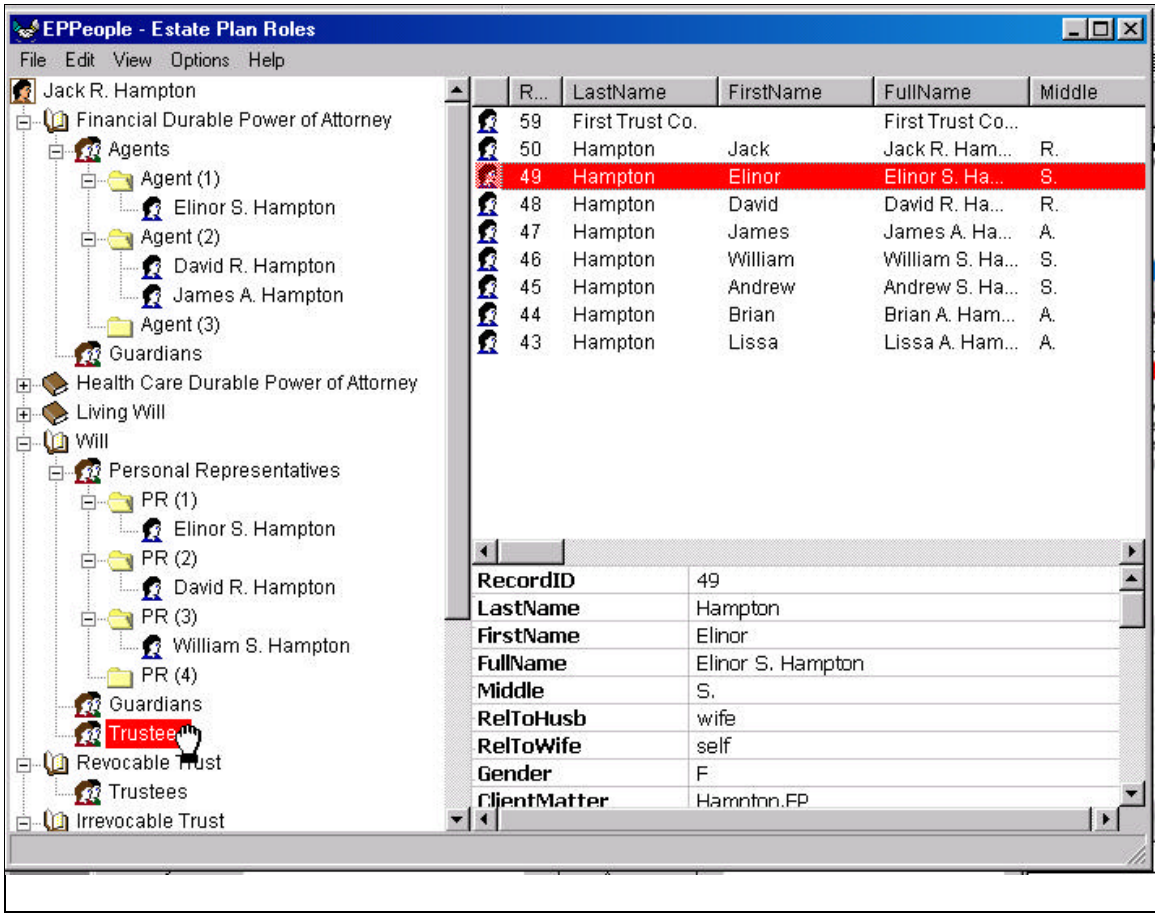
Start with the Husband's documents. Click on a person on the right, drag him or her onto the role on the left, and release the mouse button to "drop" the person on the role.

(You may have to first double click on a document or role to open it up, depending on whether the roles are visible.)

You can have co-fiduciaries by dropping two or more people on any given role.

As you add a person to a role, the system will automatically keep count. That is, if you add Agent(1), the interface will open up

Agent(2), and so forth. There is no limit to the number of levels you can have, and there is no limit to the number of people you can have at any level.



Note that if you want to change the display of people on the right side of the screen, you can drag the column headings to change the order, and you can click on a column heading to sort on that column. The default is sorting by last name, but, for example, if you want to see the children sorted in order of age, you can click on the "date of birth" column to sort by age.

To assign roles, simply click on a person on the right, hold down the mouse key, drag them to a role on the left side, and release the mouse. Start with the Huband's Durable Power of Attorney agents.

b) Copying Nodes of the Tree

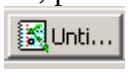
Note that you can copy just about any node on the tree. Click on the husband's FDPOA agents, for example, and copy. Then click on the husband's health care power of attorney, right mouse click, and paste. Or you can simply click on the Financial DPOA, copy,

click on the Health care DPOA, and paste. See the full EPPeople manual for more information.

First use copy and paste to copy fiduciaries, as desired, for each of the husband's documents.

Then, click on the husband. Copy. Then click on the wife, and Paste. You'll get the complete mirror image. Then you can make any manual changes to the wife's fiduciaries.

7. SAVE

When you have the data the way you want it, press Alt Tab or click the WinDraft checklist icon on the task bar,. 

and save your data file. Saving the WinDraft data file will also save the role information from EPPeople in the same data file.

8. ENTER OR EDIT OTHER DATA ON THE CHECKLIST.

(See QuickStart to Using WinDraft for more information on steps that are not related to people.) Save again, as appropriate.

9. BLAST OFF!

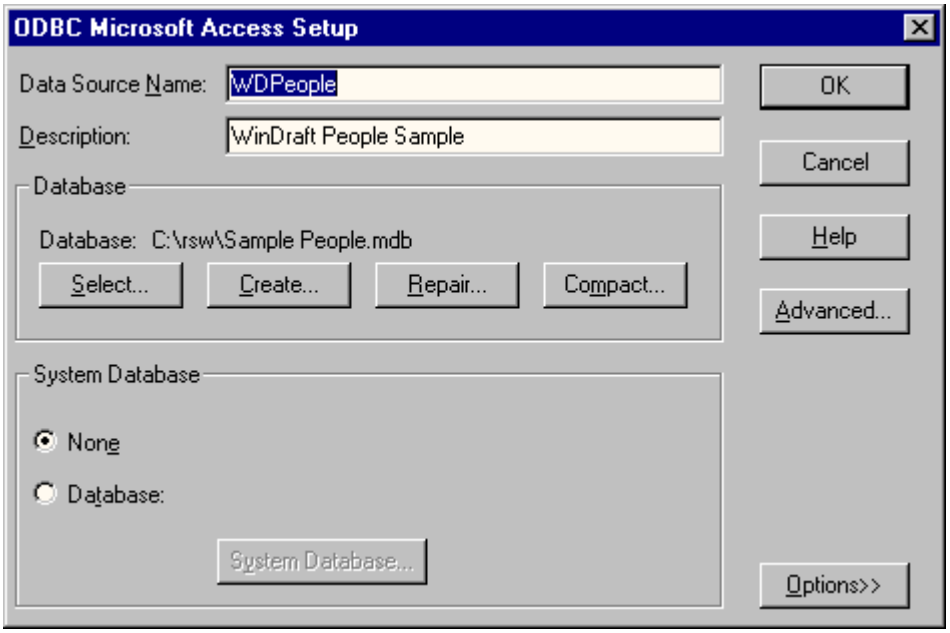
WinDraft Database Tutorial

1. Start with a database

We will assume here that you are starting with a database that you or someone else created earlier. A sample database used for tracking people is provided, and will be used in the examples following.

Before you start working with your database in WinDraft, you need to register your database with the system as an ODBC Data Source. G to Start | Settings | Control Panel, and double click on the "ODBC Data Sources" control panel. This will bring up a tabbed dialog box. Click on the "System DSN" tab, and click the "Add" button. You will be prompted for a driver. Choose the driver supporting your database (Microsoft Access, SQL server, or whatever you may be using).

A dialog similar to the following will come up. (The exact dialog depends on what sort of database you are using)



Supply whatever information the driver needs to connect to your database (in this example, just click "Select" and choose the pathname for Sample People.mdb).

Next, enter in the Data Source Name. Remember this, because this is how you will refer to the database from here on out. In our example, we will use the data source name of "WDPeople".

Click OK in the Driver Setup dialog, then click OK again in the ODBC Data Source Administrator. You are now ready to access the database from WinDraft.

2. Checklist Template

Databases are used to keep lists of people, items, or other entities. Sometimes it is enough to work with all of these together, but usually you will want to select a particular person or item for use in your output documents. This section will show you how, using the WDPeople source set up above as an example.

From Microsoft Word, if WinDraft is not loaded, load it by clicking the button or by pressing Ctrl+Alt+W. Open the Checklist Template document by clicking the button, as usual.

Now, take a minute to look through your checklist template. You should notice that it is divided into several sections. After some comments and other commands near the top, you will see some lines like:

OUTLINE Copyright "***EstatePlanning***"
    HEADING1 Title "Title Page"
END OUTLINE

After this, you will see a bunch of sections which state with something like **PAGE Copyright** and which end with **END PAGE**, with questions for the user of the system in between.

You are seeing that WinDraft structures the checklist template by dividing it into sections: one for the outline, and one for each page. To support our database, we will add a new section, just below the OUTLINE section (if it's not already there). Go in, and add a DEFINITIONS section, as follows:

DEFINITIONS

END DEFINITIONS

It is in this section that we will describe our database to WinDraft.

First, we need to tell WinDraft where to find our database. We will use the Data Source Name (DSN for short) we defined before here (if you're following the example, it's "WDPeople"). Add a line to the DEFINITIONS section (between DEFINITIONS and END DEFINITIONS), as follows:

DATASOURCE WDPeople "DSN=WDPeople"

This tells WinDraft that we are going to use a database which it will refer to as WDPeople, and that it can find it by looking at the DSN called WDPeople.

Next, we need to tell WinDraft what information we are going to want from the database. Below the DATASOURCE you just added, add (all on one line):

QUERY People WDPeople "SELECT Key, FirstName & ' ' & MI & ' ' & LastName FROM Persons ORDER BY LastName"

This tells WinDraft that we're making a database query, which we'll henceforth call People. It will access the database defined in our WDPeople datasource that we just defined, and it will use the SQL query which follows.

Explaining SQL is beyond the scope of this document (although there are many good references available). Briefly, however, the example above will return 2 columns, one with the "Key" field, and one with the full name (built from its parts).

Now, we can start using the QUERY in our document. Let's show how to change a picklist into a database query (without changing much functionality).

You are probably familiar with the SHORTCHOICE command. For example,

```
SHORTCHOICE AttorneyNum "Attorney Login"
   "1"  "Jim Eidelman"
   "2"  "Dan Marcum"
ENDCHOICE
```

The SHORTCHOICE command takes information in two columns and builds a picklist. It displays the second column on the screen, and uses the first column for variables. It works the same way when you use the database integration. If we changed the above to:

```
SHORTCHOICE AttorneyNum "Attorney Login"
   QUERY People
ENDCHOICE
```

then when we run WinDraft with this checklist template, it will query the database, and pull out a list of people in two columns. The first column will be numbers, used for the values of the variable, and the second column will be displayed on the screen in a picklist.

Find a PAGE section of your checklist template, and add the following:

```
SHORTCHOICE Husband.Key "Husband's name"
   QUERY People
ENDCHOICE
```

Start WinDraft Checklist, and go to this page. When you click on the "Husband's name" question, it should give you a drop-list of people from the database.

3. Model Document

Being able to pick people in the database like we did in the last section is nice, but it's not very useful unless we can bring more information than their key in the database into our output documents. We'll show briefly how to do this in this section.

Open an existing model document, or make a new one. (If you make a new one, don't forget to add a GETDATA statement at the top).

Add the following line:
[GETDB Grantor, "DSN=WDPeople", FROM "Persons", GET "*", WHERE "Key = " + Husband.Key + ""]

This will perform a query on the database (which is found under the DSN of WDPeople), getting all fields from the Persons table, and returning the record with the Key of Husband.Key (which is what we picked in WinDraft Checklist in the previous section). It will store information about this person in Grantor.*FieldName*. So in this example, where we have fields FirstName, LastName, MI, and Gender, the variables Grantor.FirstName, Grantor.LastName, Grantor.MI, and Grantor.Gender will be filled out. (More complicated queries could join with other tables and return as much information as you like).

These variables can then be used exactly like any other WinDraft variable. For example, try adding the following text below the GETDB line:

Last Will and Testament of [Grantor.FirstName + " " + Grantor.LastName]

Finally, we will put this all together, and test to make sure it works. Save your model document. Start WinDraft Checklist, go to the Husband.Key question you added earlier, and select a person. Click the Blastoff button, and select the model document you just edited. It should assemble to something similar to:

Last Will and Testament of Jim Eidelman

Getting Fancier

GETDB is a fairly flexible command. For more information than this document can provide, see the Author's reference.

Another scenario, though, might involve listing a number of people. For example, perhaps you have all the attorneys in your firm in a database, and you want to list them.

To get multiple people from the database, we add an open and closed parentheses to the end of the variable in the GETDB command. For example:

[GETDB Attorneys(), "DSN=WDPeople", FROM "Persons", GET "*"]

Note that we have made only two changes from the previous example. We have added parentheses to the variable name, and we have removed the WHERE clause.

When this runs, this will fill an array with information. For example:

Attorneys(1).FirstName="Jim"
Attorneys(1).LastName="Eidelman"
Attorneys(1).Gender="1"
Attorneys(1).MI=""

Attorneys(2).FirstName="Dan"
…

In addition, the variable Attorneys.Count will be filled with the number of people returned by the query.

This information can then be used in a FOR loop to do any number of other things. For example, to make a list, use:

[FOR I = 1 TO Attorneys.Count DO Attorneys(I).FirstName + " " +
Attorneys(I).LastName + "
"]

WinDraft Documentation
Additional Documentation and Notes
October 12, 1999

I. PERSONAL PRONOUNS AND OTHER GENDER-SPECIFIC TERMS 2

A. Personal Pronouns as "Functions" 2

B. Estate Planning – Gender of Husband and Wife for Mirror Image Documents 4

II. EASILY CHANGING BLAST-OFF DIALOGS FROM TEXT FILES 4

A. New Tabbed Interface for Blastoff 4

B. General Blastoff's BlastoffTemplate.txt 5

III. CONTEXT-SENSITIVE HELP DOCUMENTATION – ADD: 6

A. Enhancement – You don't need to create an entry in the .NFO file if you name the variable the same as the HTML file. 6

B. Using Word 97 or Word 2000 as your HTML Editor 7
1. Word 2000 and Internet Explorer 5.0 7
2. Word 2000 or 97 and any browser (Netscape, IE) 8
3. Inserting Screen Shots 8
4. Files and Directories 9

IV. USING WORD FIELDS TO UPDATE LATER DRAFTS 9

V. COSMETICS: VISUAL DISPLAY AFTER "BLAST-OFF" 10
1. PROGRESS 10
2. STATUS 11
3. DISPLAY ON/OFF 11

VI. PROGRAMMING FOR…NEXT LOOPS 11

A. Data Entry Checklists 11

B. FOR…NEXT loops in the documents 13
1. JAE Complete this… 13
2. How to set up the For I = 1 to CounterVariable loop 13
3. Whole Command on one line or in INCLUDEd file 13
4. Examples 13

VII. TIP ON PROGRAMMING MIX-AND-MATCH MULTIPLE CHOICE QUESTIONS 13

VIII. GENERAL TIPS 14

A. Windows Setup 14
1. Task Bar 14
2. Start Menu 15
3. Windows Explorer Settings 15

B. Favorite Utilities to Support WinDraft Authoring 16
1. TextPad 16
2. BeyondCompare 17
3. Hypersnap 17
4. WinZip 17
5. WISE installation program 18
6. SetIni.exe 18

WinDraft Documentation
Additional Documentation and Notes
October 12, 1999

I. Personal Pronouns and Other Gender-specific Terms

A. Personal Pronouns as "Functions"

First, a "user-friendly" explanation:

When you want to use a gender-specific term, you can put in a special kind of WinDraft variable that uses an "index" (number) in parentheses to tell WinDraft the gender and number.

We have preset the numbers with the following settings:

1 – male
2 – female
3 – corporation, trust or other "it"
4 – plural

Sample (showing male example)	1 male	2 female	3 corporation	4 plural

HeShe(1)	he	she	it	they
HisHer(1)	his	her	its	their
SonDaughter(1)	son	daughter	na	children
ChildChildren(1)	child	child	na	children
ren(1) (insert after "child")	""	""	na	ren
s(1) (make a noun plural)	""	""	""	s
vS(1) (make a verb plural)	s	s	s	""
xS(1) (possessive)	's	's	's	'
Executor(1)	Executor	Executrix	Executor	Executors

All you have to do, then, is use a variable to represent the number. You can put the variable in parentheses with the gender-specific term, and WinDraft will automatically put the right term in the document, without the need for any programming of that specific term. For example, [HeShe(Child.index)]

Examples: **Afterborn Children.** If any Individual, other than the Individual[s(child.count)] identified as my child[ren(Child.Count)] in Article I, claims to be a child of mine, …

Example	Before	After
one male trustee and one son	If my Trustee[s(Trustee.Count)], in [HisHer(Trustee.Count)] discretion, give[vS(Trustee.Count)] any property to my [SonDaughter(Child(1).Index)], it will be the Trustee[xS(Trustee.Count)] right…	If my Trustee, in his discretion, gives any property to my son, it will be the Trustee's right…
two trustees and two children	Same	If my Trustees, in their discretion, give any property to my children, it will be the Trustees' right…

In EPExpert, these are set in a RULES file called "genderterms.rul," which is called from EPStart.doc on the …\subdocs directory. If you want to see what all the settings are, take a look at that file, and if you want to add your own, feel free to do so.

In using this, we will typically use a SHORTCHOICE question on the checklist to ask for the gender of the person, that might look like this in the script (checklist template):

```
CHOICE DisenheritedChildren.Index "Disenherited Children: Number and Gender:"
   "1" "1 son – 'his'"
   "2" "1 daughter – 'her'"
```

"4" "more than one – 'their'" END CHOICE

The index might also get set in a RULES file SET command, or in a FOR…NEXT loop data entry screen that counts how many have been entered, as in number of trustees, number of children, etc.

B. Estate Planning – Gender of Husband and Wife for Mirror Image Documents

These variables are set in GRGender.RUL and GRGender.doc for the husband and wife. See _____.

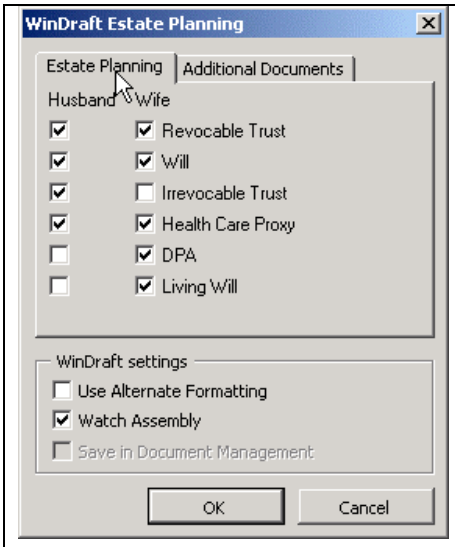
II. Easily Changing Blast-off Dialogs from Text Files

Creating the custom Blast-off dialog boxes produced great results, but takes a significant amount of time, and must be done by a "real programmer." In response to this, we have created the ability for the general and estate planning dialogs to read a text file for the information that is displayed on the screen. You can edit them with Word, Notepad, or any other editor. No ini file settings are required.

Both text files are placed on the practice system directory, such as C:\windraft\EPFirmName or N:\windraft\Reloan.

A. New Tabbed Interface for Blastoff

Here is the new estate planning blastoff. First tab contains the standard estate planning documents. Like the original one, it is very convenient for selecting mirror-image documents for husband and wife, and inapplicable documents will be grayed out.	Setting up what documents are associates with each check-box are set in the [EPXX-Documents] section of the WinDraft.ini file, and can easily be changed. Estate Planning Blastoff's EPDocsTemplate.txt



You can change the text displayed on the estate planning tab by creating a text file called EPDocsTemplate.txt on the practice system directory.

Here is a sample:

```
"WPOA","Health Care DPOA"
"WDPOA","Property DPOA"
"WlivWill","(Reserved)"
```

The entry on the left is the WinDraft.ini [EPXX-Documents] setting for the document on that line, and the entry on the right is the text to be displayed.

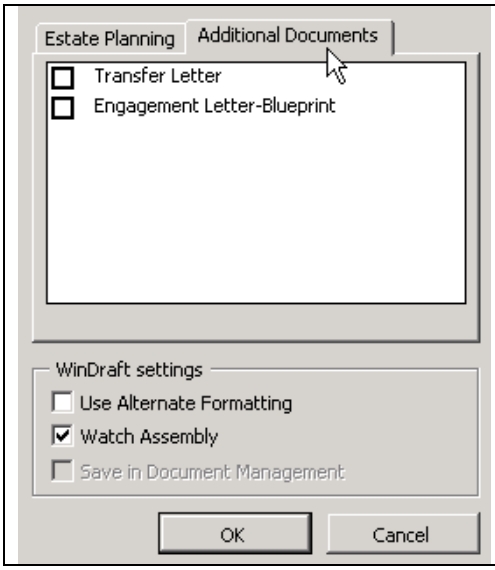
The text file can contain two kinds of entries – check boxes and tabs. This allows you to create a tabbed interface that has any documents listed on any tabs, as you choose, and to display any text you want.

(Limitations: No IF…THEN… logic is supported, no graying out of entries, and each document entry, including text that is a label, has a check box next to it. Also, no remarks are allowed.)

B. General Blastoff's BlastoffTemplate.txt

For the general blastoff, (macro named BlastoffGeneral) or estate planning blastoff (BlastoffEstatePlanning), simply create a text file on the practice system directory, and name it "BlastoffTemplate.TXT".

The general blastoff, which can be used for litigation, banking, or any other application, or the general part of the estate planning interface, looks like this:



To create a Tab, like a tab on a file folder, simply use a "NEWTAB" line, with a comma between the NEWTAB command and the label text to be displayed on the tab.

For each document on a tab, enter the name of the Word document or template that gets merged when the user selects this item, and the text to be displayed to the right of the checkbox for the item. Each is in quotation marks, and separated by a comma.

The text in the BlastoffTemplate.txt file looks like this (note: the following example does not exactly correspond to the sample displayed above):

```
"NEWTAB", "Additional EP Documents"
"FirmStandardInvoice.dot","Firm Standard Invoice to Clients"
"TransferLetter.dot", "Transfer Letter"
"Blueprint.dot","Engagement Letter-Blueprint"
"NEWTAB","Corporate"
"ArticlesOfIncorporation","Articles of Incorporation"
… etc.
```

Notes: The program that reads this file does not have much error-handling built in: If it can't find a file that is listed in the text file, it will give an error message and not display that line. Also, there are some places where an extra space, missing quotation mark, etc., will cause a problem, so be sure to uste th correct syntax.

For the general blastoff, this will provide the complete interface and list of documents.

For the estate planning blastoff, this will provide additional documents to supplement the standard estate planning blast-off screen, which lists checkboxes in a column for husband and wife for standard trusts, will, and powers of attorney.

III. Context-sensitive help documentation – add:

A. Enhancement – You don't need to create an entry in the .NFO file if you name the variable the same as the HTML file.

Beginning with WinDraft 2000, you no longer need to put an entry in the .NFO text file for a question if the HTML help file has the same name as the variable or outline "page."

This means that all you have to do if the name is the same is to create a .HTM file on the \windraft\PracSys\htmlhelp directory that has the same name as the variable, and WinDraft will automatically find it. For example, if the outline page is named **PowersOfAttorney**, simply name your help file **PowersOfAttorney.htm.**

JAE Note: What about conflicting names in variables and outline pages?

What WinDraft will do when you click the legal help button is:

1. See if there is an entry in the .NFO file. If so, it will use it.
2. If it didn't find an entry in the .NFO file, it will look on the htmlhelp directory to try to find an .HTM file that has the same name as the variable or page.

B. Using Word 97 or Word 2000 as your HTML Editor

You can use any HTML editor you want for your context-sensitive help pages. We have usually used FrontPage, Netscape Communicator, etc.

Lately, however, we have started using Microsoft Word. This is the best bet because:

- You are already using Word for WinDraft programming, and you don't have to change programs to work with the help files.

- Also, if you are pasting text from your documents into the help screens, they can have exactly the same formatting in the help screens as in the legal documents.

1. Word 2000 and Internet Explorer 5.0

If you and the users you are creating help for all use Office 2000, you can use Word's new feature that Microsoft calls "round-tripping." You can save the document as a Web page, and Word creates a Web page that has all the Word formatting information embedded in the Web page using XML commands. You can then open the Web page in Word, edit, and Save as HTML as many times as you want.

The only caveat is that you have to be careful about moving the subdirectories when you move the file, because graphics and certain other

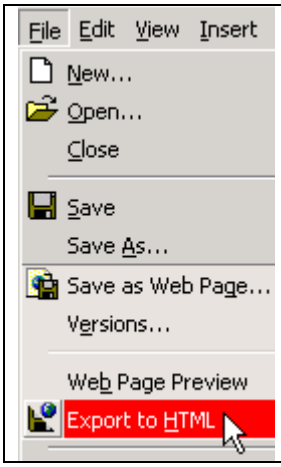
	information are automatically put in subdirectories.

2. Word 2000 or 97 and any browser (Netscape, IE)

Export to HTML. Use this approach if:

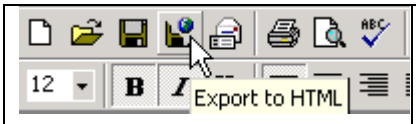
- You have Word 97, not 2000, or the users you are setting up help for don't have Office 2000.
- Your users want to use Netscape, IE 4.0, or other non-IE 5.0 browsers.
- You want to maximize for speed and leave out all the Word formatting information out of the HTML files that Save As HTML puts in the file.

Like the Save As HTML in Word 2000, it is "WYSIWYG – What you see is what you get," unlike the earlier Save as HTML in Word 95 and Word 97. It looks in the Web browser exactly the same as it does in Word.



Microsoft publishes an HTML Filter for Microsoft Word that can now be downloaded for free from www.microsoft.com/word. After you download it and run the setup program, the "Export to HTML" button will appear on your toolbar.

You edit your files using Microsoft Word, save them as a DOC file, then:



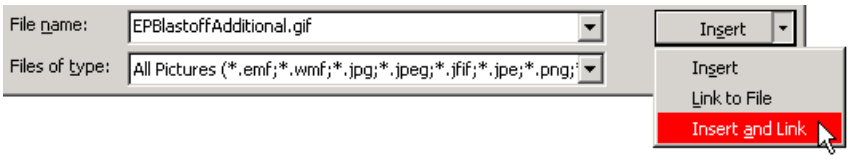
To export your DOC file, select "Export to HTML" on the File menu or click on the "Export to Compact HTML" button on the standard toolbar.

The \windraft\PracSys\htmlhelp directory will contain two files for each HTML file. They will have the same name, except that one will have a .DOC extension, and one will be named .HTM.

3. Inserting Screen Shots

If you have screen shots, I have been using the following method (used for the graphics above):

1. Use Hypersnap or a similar program to save the graphic on the HTMLHELP directory as a GIF file.
2. To insert the graphic, Insert|Picture, From File, and when you click on the GIF file, at the far right side of the dialog box is an option to insert File, Link, or File and Link.



If you click "Link to File," the link will use the same file when you export to HTML. (That's good, but the disadvantage is that if you email the document without the GIF, the picture won't be there.)

If you click "Insert and Link," When you Export to Compact HTML, it will export the graphic with a new, numbered file name, which I don't like as well. Either way, if you use a link, and if you transfer the file, the graphics will go with it, but you also have the other benefits of externally linking the file (true Web-compatible format, and if you change it, you don't have to reinsert the new picture – just click F9 to refresh the link.)

4. Files and Directories

Note that Word and FrontPage often create a subdirectory for each html page, and put graphics and other information on that subdirectory. For this reason, be sure to copy the subdirectories if you are copying the HTMLhelp information from one system or place to another. Using Word Fields for Later Drafts and Negotiated Deals

IV. Using Word Fields to Update Later Drafts

We usually think of WinDraft as a program to create first drafts of documents, and that was our design goal when we created it in Microsoft Word. While the benefits of this approach clearly outweigh the disadvantages in most situations, the problem with it is that if you are handling deals that are negotiated and go through significant changes, you may want to merge the documents a second or third time, or even more.

Here are two examples:

- One firm used forms for commercial mortgage transactions that did not change in language, but did have fill-in dates and amounts that changed several times before the closing. We used WinDraft to create the Word fields, and wrote a macro in which the fields would be updated in second and subsequent drafts with the new amounts.
- Bill Lawrence, a real estate attorney at Holland & Hart, came up with a clever way to handle leases that are negotiated after the other side reviews his WinDraft-created first draft. Not only does he use Word fields for fill-in variables that change, but he handles commonly negotiated items like this: He uses WinDraft to "blast off" two documents, sort of like the old WordPerfect primary and secondary merge documents. The first has the static information in it. The second is merged from WinDraft into Word fields, and the first document pulls in the Word fields from the second. He knows ahead of time which typical

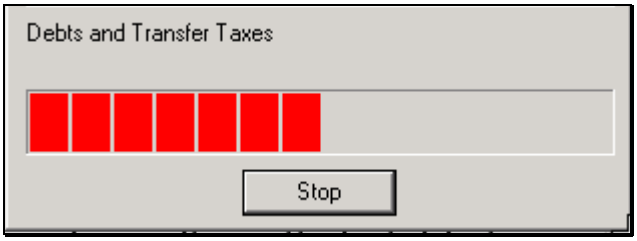
"landlord-oriented" options will need to be backed down into neutral or even tenant-oriented options, so he can later change those options this way.

V. Cosmetics: Visual Display after "Blast-off"

Users enjoy watching the document merge on the screen, and there are several things you can do to improve that experience.

1. PROGRESS

It is very important, especially when merging long documents, to give the users a cue as to how long the process will take. WinDraft has a status screen that includes a "Progress" bar and status message.



The status bar moves across the screen as the number contained in a progress command increases from 1 to 100, reflecting the percent completed. We can't make it fully automatic because WinDraft never knows how far along it really is. (You might INCLUDE a giant document at the end when it "thought" it was almost done.) You can either hard-code the command, like this: *[PROGRESS "50"]*, or make it semi-automatic using a RULES file. We have done this in a recent estate planning system by creating a file called "PROGRESS.RUL" that is called 10 times during the document. It uses the WinDraft ADD command to add 10 to a variable if it is a will, and 5 to the variable if it's the trust, a much longer document. It looks like this:

```
IF DocumentType = "POW"
      ;; POW – Pourover will – calls RULES
      ;; "Progress.rul" 10 times.
      SET ProgressIncrement = "10"
ELSE
      ;; Trust – call this rule file 20 times
      SET ProgressIncrement = "5"
ENDIF
SET ProgressCounter = ADD(ProgressCounter,ProgressIncrement)
PROGRESS ProgressCounter
```

To use it, just sprinkle the command [RULES "Progress.RUL"] through your model documents.

2. STATUS

The STATUS command will display a message, like the one shown above, on the Word status bar and above the progress bar on the Word status screen. We like to insert a status command after each major heading as a visual cue to what is going on. With IF statement programming, you can even get quite sophisticated in your messages.

3. DISPLAY ON/OFF

People like to watch the text scroll by and change, but they don't like to watch complicated programming and internal comments. Use *[DISPLAY ON]* and *[DISPLAY ON]* to show only what you want the users to see.

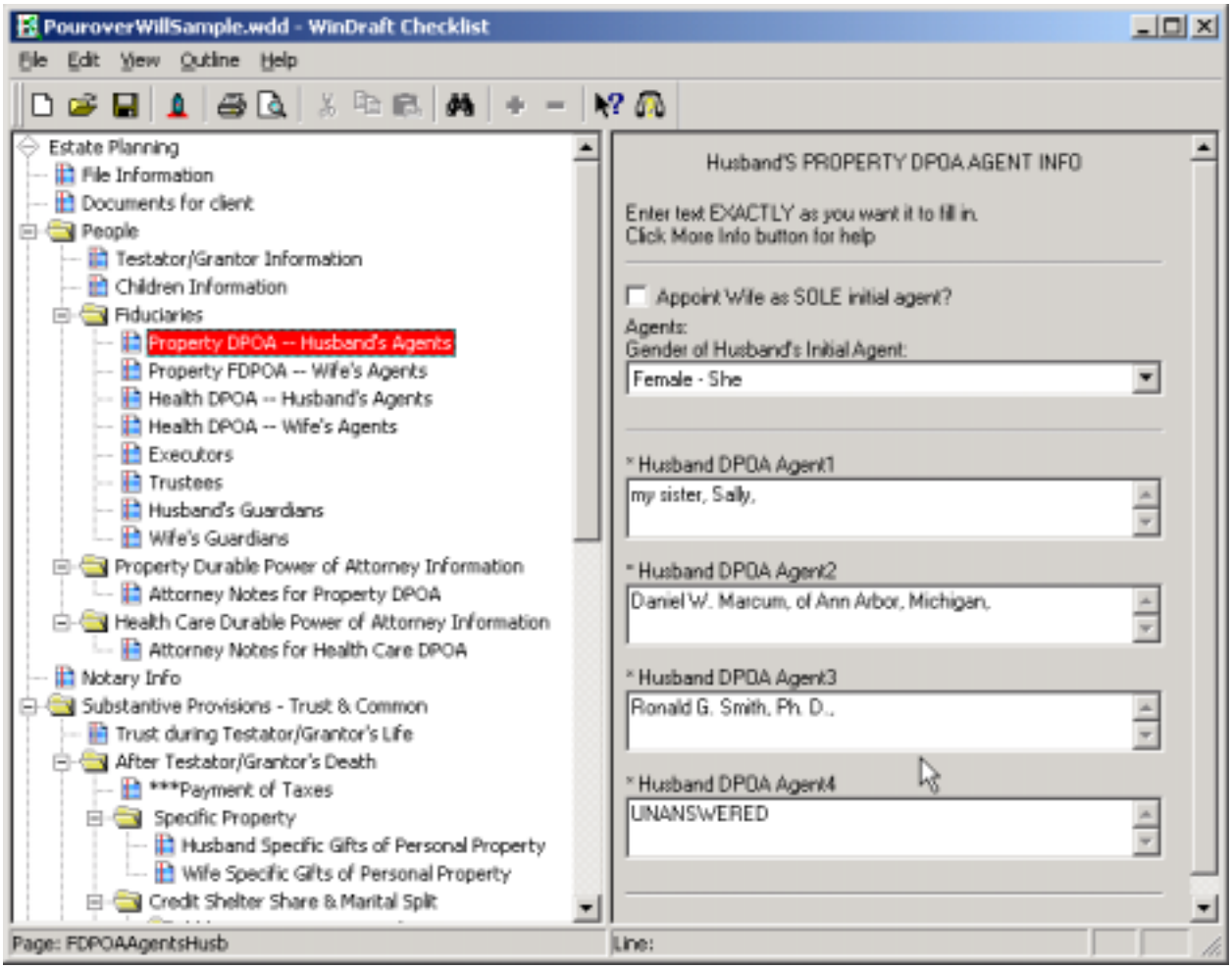
VI. Programming FOR…NEXT Loops

WinDraft has a repeat loop structure in the form of a FOR…NEXT loop.

(If the following seems confusing to you, don't feel bad. Research at the American Bar Foundation found, when teaching lawyers to program, that lawyers and law office staff members had an easy time with IF statements and subroutines, but for some reason had an unusually tough time with repeat loops.)

A. Data Entry Checklists

In many cases, you don't know ahead of time how many of a repeating item you will have, and you don't want to have to do a lot of programming for each of the items. For example, in handling fiduciaries, you might have from 1 to several initial and successor fiduciaries. The best way to handle this, when you can, is to set up a repeat loop. In this way, you only have to create a prompt for each of the repeating variables one time. As there become more and more variables within that item's information (e.g. name, gender, age, address), the benefit of being able to reuse the same programming in a repeat loop makes it much easier. Here is what a simple example looks like, for agents 1 through 4 in the husband's durable power of attorney:



The programming to create that repeating data looks like this: (I have shortened the variable names for simplicity.)

```
FOR Agent.Count = 1 TO 4
        LONGTEXT 2 Agent(Agent.Count).Names "* Husband DPOA Agent"  +.Agent.Count
ENDFOR
```

You can put just about any logic you want between the FOR… and END FOR commands, including a number of variables, IF…ENDIF commands and SET commands.

B. FOR…NEXT loops in the documents

1. **JAE Complete this…**
2. **How to set up the For I = 1 to CounterVariable loop**
3. **Whole Command on one line or in INCLUDEd file**
4. **Examples**

    a) Inserting a list of people

    b) Calculating a variable that contains a list of people with "comma logic"

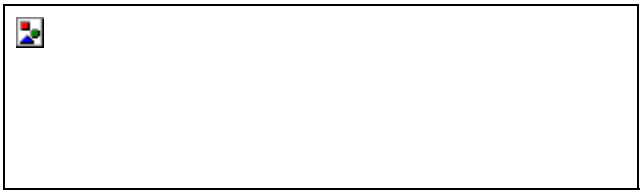
    c) Counting

VII. Tip on Programming Mix-and-Match Multiple Choice Questions

If you need logic in the documents that may include different text depending on options for whether one, none or both alternatives are selected, we prefer not to use a YesNo checkbox for each in order to keep the logic in the documents simpler. (This avoids needing to program IF Will = "Yes" and Trust = "No…, and all the other combinations.)

In the documents we use the "C" operator (Contains) to include the paragraph if the document contains "will" or "trust", as appropriate, and you don't need any and-or-not logic.

You can set it up as radio buttons, like this, using the CHOICE command:



Or you can set it up as two checkboxes, using the CODE command, like this:

The scripting of the Checklist template/script for the above is:

```
LABEL "Pers effects and residence questions show"
```

LABEL " 2 ways to mix and match multiple choices"
LABEL "Do you like radio buttons or check boxes better?"

CHOICE DistributePersEffectsUnderWillOrTrust "Provide for distribution of personal effects under Will or Trust?"
"Will" "Will"
"Trust" "Trust"
"WillTrust" "Both Will and Trust"
"" "Neither Will Nor Trust"
END CHOICE
CODE DistributeResidenceUnderWillOrTrust "Provide for distribution of residence under Will or Trust?"
"Will" "Will"
"Trust" "Trust"
END CHOICE

VIII. General Tips

by Jim Eidelman

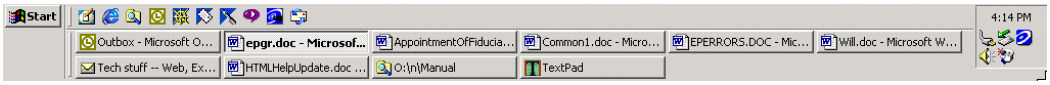
A. Windows Setup

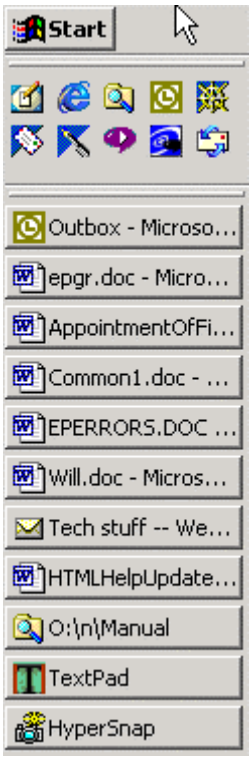
1. Task Bar

In Word 2000, Word gave each open document its own full window. It looks like there is a separate version of Word open for each document, but there isn't. At first I hated this because I usually have a lot of documents open, and the task bar at the bottom of the screen had buttons that were so narrow that they were unreadable, like this:

There are two ways to get your toolbar back so that you can read what the buttons represent:

- Drag the task bar up so that it displays 3 or more rows. It will look like this:





- Drag the task bar to a vertical position on the left or right side of the screen. This is my preferred approach. I hated it at first, but after a few days I got used to it and like being able to read the full text for each open program or document, with about 25 open tasks and files displayed on my XGA laptop screen.

  Having it on the side, with start menu at the top, is disconcerting at first, because you have to go to the top to click start, then the bottom to click "Programs." I started using Control-Escape instead of clicking Start, and it makes it quicker.

Either way, side or multiple rows on the bottom, you will probably want to click "Autohide" on task bar properties. This makes the task bar hide when you aren't using it, and it pops up when you move the cursor over to that edge of the screen.

This is a must
-- If you have a small screen
-- You use the recommended approach to move the task bar to the top or expand to several rows on the bottom.

-

2. Start Menu

I believe that you should be able to run the programs you use most often with one or two mouse clicks. Therefore, I hate it when programs I use most are buried several layers deep inside, for example,

Start
Programs
Microsoft Office
Microsoft Word

All you have to do is drag and drop shortcuts for your favorite programs onto the Start button, or go into the Advanced button under task bar properties, and make any changes to the start menu you want.

3. Windows Explorer Settings

a) Turn on display of file extensions

By default, when you install Windows, it turns off display of file name extensions. It makes it hard to distinguish between WinDraft.dot, WinDraft.ini, and WinDraft.wll when you are looking at a list of files. The first thing I do when I sit down at a

machine that is set this way is go into Windows Explorer, and change the options so that you can see the full file names.

b) Show Directory Name in Task Bar

While clicking on "show file extensions" above, I also turn on the option to show the path of the folder/directory in the windows title bar.

c) "Details view"

View files in directories in "details" view, instead of a row of folder and files. This way you can see file size, date, etc., and click at the top to sort by name, date, size, type, etc.

B. Favorite Utilities to Support WinDraft Authoring

1. TextPad

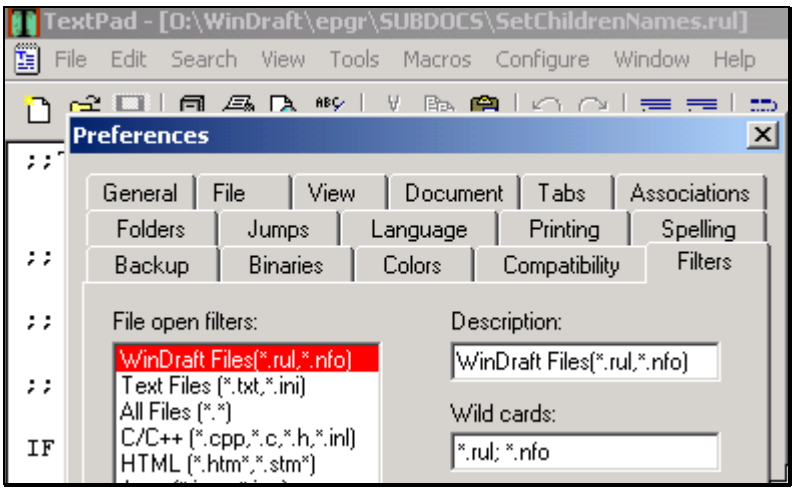
Shareware. Best text editor to replace Notepad. I use it for batch files, ini files, WinDraft rules and NFO files, and to perform batch operations on HTML files. Highly recommended. www.textpad.com

Here are the settings I use when working with WinDraft: (The tabs are under on the Configuration menu under "Preferences"):

1. Compatibility tab, I changed to keystroke compatibility with Microsoft products.

2. Under file associations, I added ini, bat, txt, and rul (WinDraft rules files), so that when you double click on these files, they will be loaded with TextPad.

3. Under "Filters," I added Windraft Files (*.rul, *.nfo) as files to be displayed when you select File Open, and I moved it up as the first choice. This is very convenient for editing rules files.



2. BeyondCompare

Shareware. Beyond Compare is my favorite program to compare and synchronize file directories. It is particularly useful when a team of people is testing and updating many files, such as in editing a WinDraft practice system. We do development and testing on a local hard disk or test directory on the network, and we store the latest good versions on a different directory. With BeyondCompare, you can compare the files in a side-by-side list, with color coding for differences of time stamp, size, etc., and buttons to move files between the left and right panel. Highly recommend.

3. Hypersnap

Shareware. All it does is grab screen shots, but it does it simply, easily, and very well. It can grab a while screen, a window, or a region. It can save in bmp, gif, and jpg formats, with control over number of colors, transparent gifs, and percentage compression for jpg. And it can crop the images. http://www.hyperionics.com/

4. WinZip

Most of the time I use WinZip to create and open zip file files. (I haven't compared it with PKZip for Windows or the other Zip programs. I just got in the habit of using it when it was the first Zip program for Windows, and I have kept using it. A Discussion of self-extracting zips, spanning disks, and working with directory structures is beyond the scope of this note, but WinZip handles all of them well. (And if you use AOL and hate the problems you have getting email attachments out of the MIME files AOL downloads, WinZip will also handle those files easily. www.winzip.com When we need to create Zip files from the command line (from a batch file), we use PKZip Command Line program for Windows 2.5, similar to the old DOS PKZip. www.pkware.com

5. WISE installation program

We have been using the WISE installation program to create setup programs(www.glbs.com). WISE is like Zip in that it compresses a lot of files into one file that takes up much less space and can uncompress the information onto the right directories on your disk. But it can also create interactive dialogs, make changes to ini files and registry settings, and contain IF…Then logic. It is great for installing and updating files on the network or on a user's laptop. (InstallShield now has an easy-to-use competitor, and Microsoft has just released a free installation that we haven't tried yet. (http://msdn.microsoft.com/vstudio/downloads/vsi/default.asp) Note that it doesn't support custom dialogs or running of scripts at the end of the setup.)

6. SetIni.exe

Public domain. An old DOS program that I couldn't find anywhere on the Web, but I found a copy on an old backup. It is a command-line program, usually called from a batch file, that adds, deletes, and edits ini file settings.

WinDraft Model Document Command Summary

Command	What it Does	Example
[*variable*] or [expression]	Fills in the blank	I, [*GrantorName*], hereby ...

IF ... ENDIF
IF ... ELSE ... ENDIF
IF
ELSEIF
ELSE
ENDIF
	Rules for logic	[IF *FractionOfResidue* = "Yes"]
		Fraction of residue formula goes here.
		[ELSEIF *PecuniaryFormula* = "Yes"]
		Pecuniary Formula goes here.
		[ELSE]
		Something else goes here
		[ENDIF]

INCLUDE	Incorporates another document. (equivalent to calling a "subroutine" in programming)	[INCLUDE "TRPOWERS.DOC"]
RULES	Same as INCLUDE, but for ASCII file containing only commands	[RULES "TAXRULES.RUL"]
GETDATA	Tells WinDraft to retrieve the information you entered at the data entry screen	[GETDATA]
SET	Set value of a variable	[SET *GrantorHeShe* = "she"]
ASK	Ask the user a question during document assembly	[ASK *CoverLet* "Include a cover letter?"]
FIELD	Inserts a Word field using the given text	[FIELD "DATE * MERGEFORMAT"]
;	Programming Comment (computer ignores what follows)	[;; JAE programming note: this is a note...]
LOG	Outputs an entry to a user-defined billing or other log file	[LOG "Timeslip.log", JAE,12/2/95,JonesCorp,0029,2.5, Drafted asset purchase agreement"]
SYSLOG	Outputs an entry to the WinDraft system log file	[SYSLOG "Asset purch agmt drafted by " +Author]
STATUS	Displays message on status bar	[STATUS "merging " + *GrantorName* + " Trust"]
DISPLAY	Turns display of document merge on or off if Watch Assembly is turned on	[DISPLAY OFF]
PROGRESS	Sets the progress meter to the indicated percentage as the document is merged	[PROGRESS "50"]
DOCTITLE	Displays document title in Windows title bar during merge.	[DOCTITLE "Loan Agreement"]
PROFILE	Outputs a profile string for Word or DOCS Open document summary	[PROFILE DocTitle = "Bill"]
HALT	Immediately stops document merge--used for debugging	[HALT]

Function	What it does	Example
SPELLOUT()	Spells out the value of a numerical variable, e.g., to convert 35 to "thirty five"	[SPELLOUT(*MinimumAgeForDist*)]
UC() or LC()	Upper case: converts "Andy Reynolds" to "ANDY REYNOLDS"; Lower case: converts it to "andy reynolds"	[UC(*GrantorName*)] [LC(*GrantorName*)]
PC()	Proper Case: changes "personal representative" to "Personal Representative"	[PC(*GrantorName*)]
REF()	References Word table, field or bookmark	[REF("TrustWill")]
MERGEFIELD()	Gets a Word Mail Merge Field	[SET *LastName* = MERGEFIELD("LastName")]
Operator	**What it does**	**Example**
NOT	Works with IF to test if something is NOT true	[IF NOT *GrantorMarried* = "Y"]
AND	Compound logic	[IF *GrantorMarried* = "Y" AND *Children* = "Y"]
OR	Compound logic	[IF *GrantorMarried* = "Y" OR *Children* = "Y"]
=, <, >, C	Comparison operators (equal, less than, greater than, contains)	[IF *MaritalTrustCode* C "QTIP"]

See User Manual for complete listing of commands, functions and operators and more complete explanations. wdcmdsum.doc 9/29/96

WinDraft Data Entry Screen
Author's Quick Reference -- Examples

CODE *Trustee* "Husb. Trustee Type Codes"
"*Grantor*" "*Grantor*"
"*Spouse*" "*Spouse*"
"*Corporate*" "*Corporate*"
"*Other*" "*Other*"
END CODE

LABEL "Gifts of Tangible Personal Property"

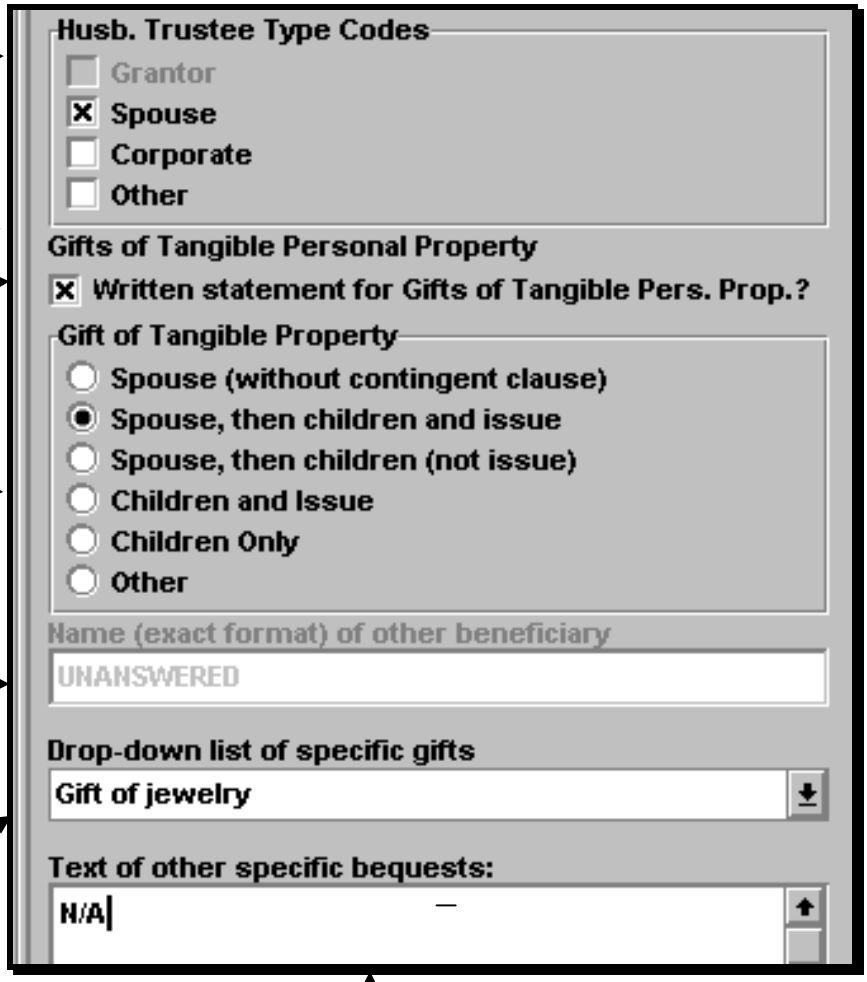
YESNO *WrittenStatement* "Written statement ..?"

CHOICE *Tangible* "Gift of Tangible Property"
"*SpouseNoCC*" "Spouse (without contingent clause)"
"*SpChildIssue*" "Spouse, then children and issue"
"*ChildrenIssue*" "Children and Issue"
"ChildrenOnly" "Children Only"
"*Other*" "Other"
END CHOICE

IF *Tangible* = "Other"
TEXT *OtherName* "IF 'other,' name of other beneficiary"
ENDIF

SHORTCHOICE DropDL "Drop-down list..."
"Jewelry" "Gift of jewelry"
"Baseball Glove" "Baseball Glove"
END SHORTCHOICE

LONGTEXT 3 *BequestList* "Text of other specific bequests"



Other commands that display: LINE – Displays a horizontal line.
EDITCHOICE – Same as SHORTCHOICE, but lets you type in data not on list.
SPACE – Extra blank line
TITLE – Same as LABEL, to display text, except that it centers the text.

Other commands you can't see on the screen:

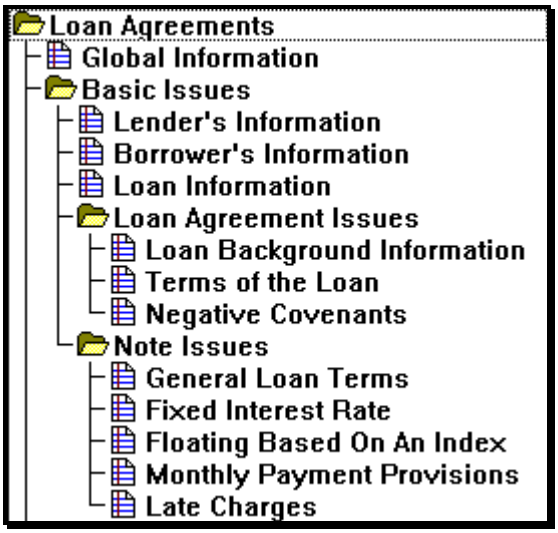
IF…ELSE…ELSEIF…ENDIF	Used to "gray out" questions that don't apply or to control the flow of logic. (See TEXT example above.) Also hides outline headings
SET	Used to set an answer, such as: SET State = "New York"
FOR…NEXT	Repeat Loop

AUTHORING SCRIPT OUTLINES -- Command Summary and Examples

Sample Script with Outline Commands

```
OUTLINE Page1 "Loan Agreements"
HEADING1 Global "Global Information"
HEADING1 BasicIssues "Basic Issues"
  HEADING2 LenderInfo "Lender's Information"
  HEADING2 MakerInfo "Borrower's Information"
  HEADING2 LoanInfo "Loan Information"
  HEADING2 LoanIssues "Loan Agreement Issues"
    HEADING3 Background "Loan Background Info"
    HEADING3 TermsOfLoan "Terms of the Loan"
    HEADING3 NegCovenants "Negative Covenants"
  HEADING2 NoteIssues "Note Issues"
    HEADING3 GenLoanInfo "General Loan Terms"
    HEADING3 FixedInterestRate "Fixed Interest Rate"
   HEADING3 Floating "Floating Based On An Index"
   HEADING3 Payment "Monthly Payment Provisions"
   HEADING3 LateCharges "Late Charges"
END OUTLINE
```

Left side of data entry screen it creates



Commands with Examples	***Explanation***
SYSTEMNAME "Loan Agreements"	• System Name command. This name will display when the user clicks on "Help|About WinDraft Data Entry"
COPYRIGHT "Jones & Smith 1995"	• This will display a special Copyright notice for the practice system both on the system's title page on the right side of the screen and when the user clicks on "Help|About WinDraft Data Entry"
;; LoanAgmt.doc -- JAE last edited 7/9/99.	• Comment (WinDraft ignores it) so that when people read the script they have some audit trail/explanatory information.
OUTLINE Page1 "Loan Agreement Data"	• The on-screen outline MUST begin with the OUTLINE command, followed by Page1, and the text you want displayed next to the icon of a folder at the top of the outline.
HEADING1 Parties "Info about parties" **HEADING2** Lender "Lender Information" HEADING2 Borrower "Borrower Info" **HEADING3** PrimaryBorrower "Primary" HEADING3 Guarantors "Guarantors" etc. etc.	• Each outline heading command line must have the following elements: 1. Heading level command (will be indented based on number) 2. Name of data entry screen PAGE that is displayed or printed with this outline heading. 3. Text to display or print in the outline heading, in quotation marks.
END OUTLINE	• There must be an END OUTLINE command after all of the headings.
PAGE Parties TEXT variable "Prompt" YESNO variable "Prompt" CHOICE variable "Prompt" "option 1" "Text for Option 1" "option 2" "Text for Option 2" END CHOICE **END PAGE**	• Each HEADING must have a matching PAGE and END PAGE, with the name of the page exactly matching the name in the heading. Between the PAGE and END PAGE commands are the data entry prompts. See *Authoring Data Entry Screen Page Command Summary* for examples
PAGE Default LABEL "Text you want on a default page" **END PAGE**	• Each script should have a page named "default" that will be displayed on the right side of the screen for any heading that doesn't have a matching page.
PAGE Title LABEL "Text you want on title page" **END PAGE**	• When the script is first loaded, the title page will display on the right side of the screen. It will display the name of the system, the copyright , and any other special information you want .

When Document Automation Makes Sense

(c) James Eidelman 1997	James A. Eidelman • Eidelman Associates • 734-769-1500 • www.lawtech.com
This document was drafted to help a large law firm who had acquired a "site license" to WinDraft decide where to begin in implementing it.	

Software for computer-assisted document automation is an add-in to word processing. It runs the gamut from what people in the artificial intelligence community would call "expert systems" down to what Mark Hellmann calls "fancy mailmerge."

Document Automation software, such as WinDraft, HotDocs, Scrivener, and PowerTxt, typically differs from the word processor's merge and macro facility in the following ways:

	Document Automation	**Word Processing Macros and Merge**
Organized and interactive way of gathering the facts and making legal decisions.	Yes	No
On-line context-sensitive help regarding legal issues.	Yes	No
"Smart Forms." Power to change text automatically, based on the facts and data entered. (IF...THEN... rules are often embedded within the document in a lawyer-friendly way, or other approaches organize logic in a way that can be understood by lawyers and secretaries who are not programmers.) I like to say that the computer can "think like a lawyer."	Yes	No
Saves the answers (data) in a way that can be reused for future drafts and with other documents.	Yes	No

Can print out the data in a lawyer-friendly, auditable way.	Yes (some)	No

	***Strong* Need for Automation** (Buy or build a "document assembly" application)			***Weak* Need for Automation** (Use a good form-bank and document management instead)		
Criteria	**Description**	**Examples**		**Description**	**Examples**	
		Transactional	**Litigation**		**Transactional**	**Litigation**
Speed*	Need ***fast*** turn-around* (See quote below)	Business acquisition, offer to purchase	Emergency restraining orders	No hurry	Pension Plans, Employee Policy Manuals	Civil Damages Complaints
Volume	Reasonably ***large volume*** of matters that are ***similar*** enough to justify use of the system.	Qualified plans (especially when the law changes), Business Acquisitions, Loan Agreements, Condo development, License Ageements	Specialty litigation (same types of cases for different clients (PI, asbestos) or one client with lots of cases that are similar (jeep rollovers, gas tank storage cases, etc.)	*Infrequently* do this type of work *or* each matter is truly *unique*	Unique transactions (sale of ski resort)	Unique commercial cases (Anti-trust case against Microsoft)
"Smart Forms" will help.	Documents have enough ***complexity***, individually or as a group, that having the computer ***"think like a lawyer"*** will help.	Contracts, wills, patterns, variations on a theme, such as Standard Contracts, Wills, etc.	Standard pleadings, motions, etc.	May not be worth the trouble for documents that are simple merge documents without alternative paragraphs.		

Control of Content	The firm can ***control the first draft*** of the documents, because they are *unilateral*, *proactive*, or we create the first draft for discussion.	*Unilateral documents* (wills, trusts, pension plans, employee handbooks), *first drafts* of purchase contracts, leases.	*Pleadings that originate action* (complaints, interrogatories, motions and supporting memos)	Documents that are each highly negotiated before we create the first draft, or *reactive* documents in response to specific events or the documents of the other side.	Unique negotiated contracts.	Answer to complaint, stipulation, answer to interrogatories.
Revenue -- "Value Billing"	Beyond the billable hour. Clients pay by the transaction, ***not by the actual hour.***	Wills, Condo conversions, Business Transactions, etc.	Plaintiff's contingent fee work, Other litigation in which client agrees to "value billing"	Client won't pay for efficiency and will only pay for hours "by the clock," or involves original research with unpredictable time.	Transactions where model of time billing works better or client insists on it.	Much insurance defense litigation
Cost	System can be **acquired** from a publisher or **built** with in-house staff in a way that is affordable (or by consultants if cost-justified)			System can only be built by partner-level experts, and no partners are interested. No programming-literate staff.		
Need to capture "Institutional Knowledge"	Firm wants or needs to "bottle" the knowledge and skills of an expert.			Everyone in the firm knows how to do it, (or nobody does).		
Need to lower cost	***Need to lower cost*** of handling	Wills and trusts,	Volume of simpler cases involving	Work that requires	High-leverage,	High-leverage litigation at

	transactions	routine transactions	smaller amounts, pressure from competing firms or clients.	"hands-on" work by a partner	high hourly rate transactional work.	high hourly rates. (High on Bill Cobb's "value curve.")
Legal Work is Document - Intensive	Legal work product results in ***word processing documents***	Contracts, wills, deeds	Pleadings, motions, written discovery	Legal work that is handled verbally or without much paper output.	Deal making, number crunching, counselling.	Trial work, depositions, negotiations, research

* Regarding speed, an article in the Wall Street Journal entitled **"Pushing the Pace"** pointed out that "speed" has replaced "total quality management" as the new mantra of business.

> ***"The latest big thing at many companies is speed, speed, speed...It's not the big companies that eat the small; it's the fast that eat the slow."***

If your clients are into "speed, speed, speed," then document assembly technology is the only way to deliver complex documents fast and still get home for dinner.

2 7 TIPS for a Successful Document Assembly Project

Two veterans of document drafting system creation share their insight in hopes of making your future undertaking as painless as possible.

By James A. Eidelman and Christopher Maeso

A computer program that can draft complex legal documents in minutes is the "holy grail" of many lawyers who handle business transactions. After all, those with document assembly systems can reap the benefits of higher effective hourly rates, faster turnaround time to clients, and consistency and quality in all their documents. Successful document assembly systems allow attorneys to focus on the creative, personal and fun parts of practicing law, letting the computer do the cutting and pasting.

Unfortunately, stories about failed document assembly projects are legendary, and anyone who has tried developing a system has learned too many lessons the hard way. The development of document assembly systems can become a black hole. When the system finally gets finished, if it is not flexible and user-friendly, it will fall into disuse as the law and needs of the firm change or as the people who developed the system move on to other projects.

The following tips are based on hard lessons learned. If followed, they should lead to the successful completion and implementation of a maintainable system that attorneys and staff rely on and use profitably. Whether you use Word, WordPerfect or Word Pro, and whether you use WinDraft, HotDocs, Scrivener, PowerTXT, General Counsel, ExperText or Workform, these general principles will help.

Managing the Project

1. Get Everyone Involved: If you practice in a group, make sure that everyone feels that the system is partly theirs as it is developed. (But don't let a committee bog the project down too much, either.) Involve the group in testing and improving the system as you proceed. If you are the sole attorney developing it, and it is viewed as your system, it will be hard to get others to use it — no matter how good it is.

2. Start Small and Grow Slowly: Get a small system drafting simple

and often-used documents like powers of attorney, simple wills and client opinion letters up and running first. Establish the concept of the system, test your approach and use it on a real client project.

3. Develop the System While You Do Billable Work: If you refuse to implement the system until it is 100 percent complete, you run serious risk of failure. Waiting until the system does everything you (and your committee) think will be needed to "do it right" will lead to a project that'll never be finished. Instead, add variables, text and logic with real clients in mind as you go along, each time testing the system so it correctly creates first drafts of client-ready documents. In several months you will have painlessly created a working system that meets your needs most of the time.

4. Sustain Your Efforts: It is much easier to complete a system or one of its components if you apply a sustained effort. This greatly reduces the amount of time you spend relearning how your system works. Block off development time and don't allow yourself to be interrupted.

5. Finish One Document at a Time: Complete one document before moving on. This way you know that all documents are correct, and you and your partners can see the project progressing.

6. Appoint a System Manager: Make sure there is a lawyer who takes full responsibility for the legal accuracy of the system. It is this person's responsibility to make sure that the system is kept up-to-date. In the three-ring binder you'll use to manage your system (see Tip 21), have that attorney update each section as the law changes. This way, anyone can see whether the last changes to the system include recent changes in the law. It also takes a responsible attorney to coordinate the management and use of the system in the firm. This is often necessary in ways that a paralegal, secretary, or MIS staff simply can not do.

7. Don't Try To Integrate With a Database: . . . at least at first. Trying to work on a database at the same time you set up a document assembly system is like running a marathon with a young child clamped on to your leg — you won't cover much ground. It is very difficult to know how to define the variables for the database until the document system is complete. Also, because the law is always changing, law practice systems must be remarkably flexible. Database integration can hinder this flexibility. Don't try it unless you want to change your career from practicing law to computer programming. (There are exceptions, such as looking up limited data in an existing firm database, developing commercial applications with a professional programmer, integrating relatively simple documents with an existing case management database, etc.).

Choosing Software

8. Pick a Word Processing-Based System . . . : Pick a program that works with your word processor. This prevents you from having to learn new function keys and eliminates wasted time importing, exporting and dealing with differences between the way the document assembly software works and how your word processor works. A word-processing add-in is much easier to learn and maintain. More importantly, it lets attorneys divide the task of developing a system by letting the lawyer do the lawyering while a secretary does the formatting and non-substantive editing.

9. With An Open File Structure . . . : If your whole system is in one giant file, it is virtually impossible to use the team approach. You will often want to have one lawyer work on a section of a loan agreement in the office while a second works on a security agreement in another city and a secretary works on formatting and inserting Word or WordPerfect fields in a different part of the loan agreement. With a system based on word processing documents, this is easy. You can cut, paste, e-mail and redline text to your heart's content. Some non-word-processing based systems won't support coordination of separate efforts.

10. And Good Balance of Power and Approach: Some packages use IF . . . THEN logic to program rules into a document. Others offer a lawyer-friendly way to browse through paragraphs and select the ones you want. To many, this is a religious issue between absolute good and absolute evil. The truth is that they are different tools, one a hammer and one a saw. Each lawyer will have a different preference, and each tool has advantages in different practice areas and firms.

At their best, systems that utilize IF . . . THEN rules are more challenging to create and maintain, but they are worth the trouble. These systems save not just secretary time, but also lawyer time. The law is full of rules. By using systems that let you program the rules, you can, in a small way, teach the computer to think like a lawyer.

For example:

I. "IF a beneficiary is a trustee, THEN put in language restricting the trustee's powers."

II. "IF the amount in controversy

27 Tips for Document Assembly

exceeds $50,000, THEN there is federal jurisdiction."

Systems that let you put the IF . . . THEN rules right in the text are better than those that hide them from you. While actually seeing the logic takes a little getting used to, it makes the documents "auditable," so that you can make sure the rules are being applied correctly.

Note that the key to the success of an IF . . . THEN system lies in how well the rules are stated. Use language that lawyers who are not programmers can understand. For example, a lawyer can easily comprehend a programming statement like "IF plaintiff has children" or "IF there is diversity of citizenship," but will find cryptic or awkwardly drafted logic hard to follow.

11. **Ignore Macros and Merge Programs:** Systems based on WordBASIC, field programming or the WordPerfect macro language are either too difficult for the average lawyer to understand or too limited in power. (It's a shame that word processors don't have the basic document assembly features of the old WordStar Mailmerge or Dictaphone.) Most document assembly packages combine the power you need with ease of use for mere mortals. You can get great looking macro-based systems up and running, but as the law and staff change, the firm can't maintain them. Also, as the word processors change versions, you may find yourself with an obsolete program. Just ask anyone who programmed a system in WordPerfect 4.2, then 5.0, then 5.1, then 6.1 . . .

Programming Techniques

12. **Test As You Work:** As you add data elements, logic and language, test your system every few minutes. Debugging is a part of any project, and it is much easier to do if you test as you go along.

13. **Use Descriptive Variable Names:** It is easier to understand "I, [Grantor Name], a resident of [Grantor County], [Grantor State]" than "I, [Gr], a resident of [Gcty], [ST]."

14. **Take Advantage of Your Program's Internal Comments Feature:** Most systems let you embed annotations, comments or notes. Use these to leave an audit trail, explaining both your legal and programming logic for yourself and those who will follow.

15. **Use Word/WordPerfect Fields Liberally:** If you use Word or WordPerfect fields to handle changing terminology like "he/she," "his/her," "testator/testatrix," "will/trust," "plaintiff/defendant," etc., the documents become much easier to read. With Word, you can use Tools, Options to show these fields shaded with gray and with the variable fields showing or not. This makes it much easier for a partner reviewing the forms or editing the model documents on screen to concentrate on the logic and the language. "To him in his discretion" is far more lawyer-friendly than "to [Trustee Him or Her] in [Trustee His or Her] discretion."

16. **Hide Complexity in Subroutines:** There is no way around some complex programming. When we have to do seriously complex programming in our documents, we move it to subroutines or rules files that contain "the ugly stuff." This way, attorneys who are working with the systems don't have to be distracted by the programming. This includes "IF statements" that change terminology in the middle of a paragraph, "IF statements" to insert signature lines, and the like. If there are 10 places in a document where the same logic causes alternative text to be inserted, you can put that logic in a subroutine and either call the subroutine from each of the 10 places or have the subroutines set a variable and insert the variable in each place. Whether such subroutines are called rules files, computation elements or simply included documents, most document assembly engines have some way of accomplishing this result.

17. **Color-Code Your Commands:** Use color and different fonts for commands, and use different colors for nested logic. We usually use italics and blue text for most programming commands and sometimes use red, green or magenta to make it easier to read the nested logic. For a simplified example, see Figure 1.

```
[IF Grantor Has Children]
    [IF Children Are Minors]
        I give the remainder of my property to the
        Trustee, to be held in trust for my children,
        under the terms of Article IV.
    [ELSE]
        I give the remainder of my property to my
        children, to be theirs absolutely.
    [ENDIF]
[ENDIF]
```

Figure 1: An example of using color coding to make programming language clear, as suggested in tip 17.

18. **Print on a Color Printer:** If you use color to set apart programming commands, get a color printer. The attorneys will appreciate reading hard copies of documents that have the programming commands in color as much as the author will appreciate working with color on the screen.

19. **Redline Your Documents:** Use redlining tools early and often to compare: (a) different client documents to spot special clauses that have been used under different fact patterns, (b) versions of your "programmed" model documents, and (c) documents that the system has drafted. Both WordPerfect and Word have excellent tools to compare different versions of documents, and DocuComp and CompaRite are even better. Using these redlining features will help you catch errors, improve logic and capture custom changes you've made for clients.

Staffing

20. **Use the Team Approach:** Don't have your lawyers doing secretarial work, and don't have your secretaries doing legal work. The team approach works and is the only way to have a successful implementation. Use other lawyers and support staff to help do the parts you aren't good at and test the system with different sets of data.

Documentation

21. Put The System in a Three-Ring Binder: When you start the project, create a binder with plenty of dividers and keep it up-to-date with printouts of data, data entry screens, model documents, rules files, INI files and any other components of your system. Tuck an extra floppy disk with all of the system files in the binder, and keep an extra copy of both off-site in case of fire or other risk.

22. Create a Good User Manual with Screen Shots: Make this the job of a secretary or paralegal, except for editing any lawyer-to-lawyer instructions or documentation. Also, as there are changes in the law, put references to the law or copies of cases or statutes in the back of the binder.

23. Keep Your Binder and Manual Current: Tips 21 and 22 won't make any difference if you don't keep the information up-to-date.

24. Use the Online Help Features: If a CLE course outline or article contains practical advice on selecting clauses, or if a partner wants to give instructions to associates, this is a great way to make the advice available to the right people at the right time. Although some fear that document automation systems rob young associates of the learning experience of creating complex documents "from whole cloth," a well organized checklist with online advice about the issues can be an outstanding educational tool.

Maintaining the System

25. Keep the System Up-To-Date: Each part of the system should have internal programming notes at the beginning describing the content, date and author of the latest changes. The three-ring binder containing printouts

Rules to Remember

Hoffstadter's First Law Of Programming

"Every programming project takes much longer to complete than you think it will, even after you have taken into account Hoffstadter's First Law of Programming."

— Douglas Hoffstadter, one of the "fathers" of computer science

Brian Howard's Law of Computing

"You can get it quickly. You can get it cheap. You can get it working. You can combine any two of these at the same time, but not all three."

— Brian Howard, computer hacker and friend of the author

The 80/20 Rule:

"You get 80 percent of your results from 20 percent of your efforts, and 20 percent of your results from 80 percent of your efforts. Applied to document assembly, 20 percent of the forms and clauses will be used 80 percent of the time. Make your Phase I a system that will automate that 20 percent, and let the rest be done the old fashioned way."

— Adapted Pareto Rule, from 1920s economist Pareto

Pournelle's Rule:

"Better is the enemy of good enough."

— Jerry Pournelle, Byte columnist and science fiction writer

Eidelman's Law:

"It always seems 90 percent done."

— Jim Eidelman, legal technology consultant and author of article

Software Options

A variety of document assembly/expert system solutions are currently available to the legal practitioner. Below we've listed some of the more popular, "off-the-shelf"-type programs to get your search started.

HotDocs
CapSoft
(800) 500-DOCS

HotDocs document automation software is an extremely well known category leader due to its low price ($99) and a number of tie-in relationships with other legal vendors such as Lexis-Nexis, Folio, WordPerfect, AmiPro, SoftSolutions and PCDocs. Available for both Windows and DOS systems, HotDocs can convert existing word processing files into smart templates to allow assembly of customized documents. CapSoft, maker of HotDocs, also publishes the more complex CAPS document assembly engine.

PowerTXT
Intercon
(800) 422-3880

PowerTXT is another Windows-based intelligent document assembly program. It uses color-coded outlines of model documents to create variables for the insertion of client and case information. It works with all three major word processors and has an associated program, PowerForm, for filling in forms created by any other application, including spreadsheets. Pricing is $250 per user.

FormBank
ExperText Systems
(800) 387-2625

FormBank from ExperText Systems takes a slightly different approach to document assembly by utilizing the power of Windows and allowing textual information to be deposited as forms through an object-oriented, question-and-answer menu system. FormBank is priced at $395.

WinDraft
Eidelman & Associates
(800) 77-LAPTOP

WinDraft from Eidelman and Associates is a Windows-based document assembly engine created by Jim Eidelman. An add-in for both Microsoft Word and WordPerfect for Windows, WinDraft can be used to create expert systems for drafting documents in a variety of practice areas. A single-user version of WinDraft costs $495.

— Tom O'Connor, attorney and computer consultant, San Diego, Calif.

ADVOCATE SOFTWARE

"I'M SORRY TOO, MISS BICKLENIK, BUT THE MOST WE CAN GET FOR ECONOMIC DAMAGES IS $89.43."

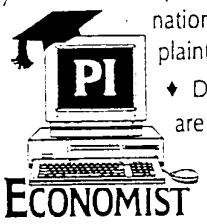
Now There's A Better Way To Calculate Economic Damages...

Put An Economist In Your Computer.

Now you can get quick assessments and expert data easily on your computer with Personal Injury-Economist and Wrongful Termination-Economist For Windows.

Get the answers you need for economic damages *immediately*. Calculate "what if" scenarios on-the-fly.

♦ Written by economists to assist the law professional. Organizes all damages in your computer.



♦ Personal injury, medical malpractice, product liability, wrongful termination or discrimination cases for plaintiffs or defendants.

♦ Detailed reports and graphs are professional, and universally respected for clarity and accuracy.

♦ Go into settlement conference fully prepared, knowing where your numbers came from. Many automatic features built-in. 800 phone support and we can assist in generating reports.

For Free Samples Call 1·800·800·1393

Evaluate Advocate Software on your computer.
Call and ask for free samples and reports.

ADVOCATE SOFTWARE PC-ECONOMIST SERIES

PERSONAL INJURY ECONOMIST	WRONGFUL TERMINATION ECONOMIST	STRUCTURED SETTLEMENT ECONOMIST
PI-Economist For Windows $299	WT-Economist For Windows $245	Structured Settlement-Econ. $195
Also Available... Wrongful Death For DOS ...and soon for Windows	Purchased w/PI-Economist $125	Purchased w/PI-Economist $125

FREE DEMO DISKETTES
– Order any or all without obligation –

·60 DAY MONEY BACK GUARANTEE· Order immediately for use with existing cases, use VISA/MC/AE
INTELLIGENT SOFTWARE FOR ATTORNEYS SINCE 1986 · 1208 ROCKLEDGE LANE · SUITE 8 · WALNUT CREEK, CA · 94595

CIRCLE 26 ON REACTION CARD

27 Tips for Document Assembly

should be initialed and dated by the lawyer who is the system manager. The system manager must track changes in the law and take responsibility for the system being legally correct. Just like the law and the firm, the system is never done and is always changing. If you don't maintain the system, it will become brittle and people will stop using it.

The Bottom Line

26. Don't Charge by the Hour: Spending money and time to create a system that lets you do more work in less time is well worth it if you can charge on a flat rate or for results. It's counterproductive if you are going to charge by the actual time spent. Change your retainer agreements so that you are doing it in an ethical way, charging for your documents, advice and results — not hours.

27. Make the Capital Investment: Business people understand the importance of capital investment. The airline with the newest fleet of planes is the most profitable, while the airline with the oldest enters bankruptcy. Hardware and software costs are only a small part of your operating budget, but the investment of your time and effort can produce an effective practice system that can help launch your firm to the top.

JAMES EIDELMAN is president of Eidelman Associates and publisher of WinDraft, a document assembly add-in for Microsoft Word and WordPerfect for Windows. Eidelman Associates is a Microsoft Solution Provider that specializes in supporting law firms and law departments in the implementation and integration of Windows-based practice solutions. Eidelman Associates can be reached at (800) 77-LAP-TOP or on the Web at http://www.lawtech.com.

CHRIS MAESO is a Michigan attorney specializing in corporate law and the development of expert systems. He can be reached over the Internet at cmaeso@milaw.com.

WORD Progress

Summer 1995 Vol. 9, No. 2

WinDraft 1.1

Commonsense Document Drafting for Windows

By Vince Wilk

A simple interface and a three-step process that lawyers will find enjoyable to use.

WinDraft is a remarkable new document drafting engine that can help you create first drafts of complex documents in minutes. The program is easy to use and offers a number of features not found in other programs. In addition, it is particularly adaptable to changes in the law.

The latter is attributable to the fact that WinDraft was created by James Eidelman, a well-known lawyer and consultant who is widely considered a pioneer in the use of computer technology in the legal community. WinDraft was first publicly demonstrated this past March at the "Document Assembly Shootout" at the ABA TECHSHOW in Chicago and was very wellreceived.

The program is an add-on to Microsoft Word for Windows and is tightly integrated in the Windows environment. (A WordPerfect version is reportedly expected soon.) The program is also compatible with most document management systems,

Turn to page 2

The Brickner Report

More Tips n' Tricks

By Barry L. Brickner, Editor

In my column in the Spring 1995 WORD Progress, *I wrote about importing legal research from a CD-ROM law library into a word processing document. The legal research program that I use is DOS-based, and I mentioned that I could not cut and paste directly into my* Windows word processing documents without making an interim file. Miriam N. Jacobson from Philadelphia read my column and wrote to me describing a method for cutting and pasting between DOS and Windows programs.

Miriam wrote that there are some DOS programs that do not permit use of the Windows clipboard (unless you actually go in and open the clipboard). She has, however, been able in most cases to easily mark, cut and paste from DOS programs to Windows and even to the same or other DOS programs.

Assuming that you are running the DOS application from Windows, by pressing Alt + Enter or Alt + Spacebar, the DOS window becomes smaller and the Windows drop-down menu appears at the upper left side of the screen. You can choose Edit and Mark to mark your selection using the mouse. Then click on the upper left corner of the screen again to bring down the menu, and select Edit and Copy. Go to the application where you want to paste, and paste using Ctrl + V or another drop-down Edit menu. To

Turn to page 14

In This Issue

5 Four Dimensions of Document Assembly *by Kenneth I. Guthrie*

8 Choosing a Document Assembly Program, Part Two *by C. Howard Thomas, Jr.*

11 Increase Your Typing Speed with Flash Forward *by Roy E. Greenberg*

12 More on Macros in WPWin *by Peter C. Scott*

13 WordPerfect 5.1 for DOS: Did You Know? *by Marjorie M. Miller*

15 Contour Wrap Graphics with WPWin *by Richard C. Belthoff, Jr.*

WORD Progress

Vice Chair, Computers & Technology Division, and Chair, Word & Document Processing Interest Groups: Bruce L. Dorner, Dorner Law Offices, 80 Nashua Rd., Londonderry Professional PK., Londonderry, NH 03053. 603/434-2230. Fax: 603/432-6473. attmail: !dornerbl

Vice Chair and Editor, Word & Document Processing Interest Groups: Barry L. Brickner, 31780 Telegraph Rd., Ste. A, Bingham Farms, MI 48025. 810/258-4930. Fax: 810/645-6645. attmail: !bricknerb

WordPerfect Windows Editor: Peter C. Scott, Mulhern & Scott, 115 Court Street Portsmouth, NH 603/433-6300. Fax: 603/433-6372.

Desktop Publishing Interest Group Chair & Desktop Publishing Editor: Richard C. Belthoff, Jr., Grier and Grier P.A., Ste. 1240, One Independence Center, Charlotte, NC 28246. 704/332-0202. Fax: 704/332-0215. attmail: !belthoffr

Document Management & Systems Editor: Jeffrey G. Wyner, Conway Marken Wyner Kurant & Kern Co., L.P.A., 30195 Chagrin Blvd., Ste. 300, Cleveland, OH 44124. 216/292-3300. Fax: 216/292-3340. attmail: !jwyner; CompuServe: 76660,1661

Windows Applications Editor: Daniel S. Coolidge, Sheehan Phinney Bass & Green, P.A., P.O. Box 3701, Manchester, NH 03105. 603/668-0300. Fax: 603/627-8121. attmail: !coolidged

WordPerfect Editor: Marjorie A. Miller, Marjorie Miller Associates, 1014 Cobbs St., Drexel Hill, PA 19026. 610/789-4730. Fax: 610/789-6883. attmail: !mmiller6

Document Assembly Interest Group Co-Chairs: Charles E. Pear, Jr., Holland & Hart, P.O. Box 8749, Denver, CO 80201. 303/295-8371. Fax: 303/295-8261. attmail: !cpear

Marc Lauritsen, Harvard Law School, Langdell 129, W. Cambridge, MA 02138. 617/495-5026. Fax: 617/495-1110. attmail: !lauritsenm

Expert Systems Interest Group Chair: Henry Koltys, 1350 Watertrough Road, Sebastopol, CA 95472. 707/829-8844. Fax: 707/824-9000. attmail: !Koltysh

Ami Pro Interest Group Chair: John Zenkewich, Law Office of John Zenkewich, 1136 Wolf Avenue, N. Valley Stream, NY 11580. 516/823-0940. Fax: 516/568-1809.

Microsoft Word Interest Group: Edna Y. Baugh, Medvin & Elberg, One Gateway Center, Newark, NJ 07102. 201/642-1300. Fax: 201/642-8613.

Managing Editor: Joan Hamby, 1449 S. Indiana Ave., Chicago, IL 60605. 312/945-3055. Fax: 312/945-3057. attmail: !hambyj

Division Manager: Jill Goldstein, ABA, 750 N. Lake Shore Dr., Chicago, IL 60611. 312/988-5706. Fax: 312/988-6281. attmail: !goldsteinj

© 1995 American Bar Association Section of Law Practice Management. All rights reserved. The views expressed herein have not been approved by the ABA House of Delegates or the Board of Governors of the American Bar Association and, accordingly, should not be construed as representing the policy of the American Bar Association.

ISSN # 1055-3983.

Editorial Policy: *WORD Progress* reports on how lawyers and legal professionals apply word processing, document production and document assembly technology in the law office. *WORD Progress* is published four times per year for members of the Word and Document Processing Interest Groups.

Article submissions and letters: Please send to Editor Barry L. Brickner, above. **To join the Interest Groups call 312/988-5619.**

Note: Rather than put a trademark symbol after every registered product name, we state that we are using the names in this publication only in an editorial fashion with no intention of infringement.

WinDraft 1.1

continued from page 1

including PC DOCS, WorldDox and SoftSolutions.

User's Perspective: Drafting Documents

I tested the program's current version, WinDraft 1.1, using Microsoft Word 6.0c on a network with Windows for Workgroups 3.11, Windows NT 3.5 and the prerelease version of Windows 95.

Those who know how to use Microsoft Word need to learn just a few additional steps to merge documents with WinDraft. In fact, the user only needs to click the mouse on three new buttons on the toolbars to create client documents. The three-step WinDraft process basically works like this:

- You select a practice system.
- You work with client data.
- You merge the documents.

Selecting a Practice System

The user first picks a "practice-system," which is a data checklist and group of documents that fit together in a practice area. It looks like Figure 1.

Working with Client Data

The most innovative part of WinDraft is the way it gathers information. After Microsoft Word is loaded, you click the mouse on the "Checklist" button and the WinDraft data-entry program appears on the screen. I particularly like the following aspects.

An outline interface. With most document assembly systems, the user is quickly lost in the mass of details and sequential questions, which make it difficult to "see the forest for the trees." WinDraft's beauty is its simplicity—the left side of the screen always shows the "forest," and the right side shows the "trees." The two sides form a client information checklist presented in a two-panel screen, as shown in Figure 2.

The left panel contains an outline of the relevant legal and factual issues and the right panel contains a dynamic data-entry screen for gathering the facts. The interface uses common Windows interface elements, such as checkboxes for "Yes/No" questions, radio buttons for "choose one of many," and text boxes for typing in names or other text. If the author has taken the time to set it up, "Legal Help" is available to explain each question or series of questions on the data-entry screen.

"Graying out" of irrelevant questions. I particularly like the way WinDraft handles logic in its data-entry screens. As selections are made that make other questions irrelevant, those that do not apply are instantly grayed out and cannot be answered. In some cases, whole screens will be grayed out, while in others just a single button in a long list of options will become unavailable. This puts the lawyer in control, never "flying blind" as with many other systems.

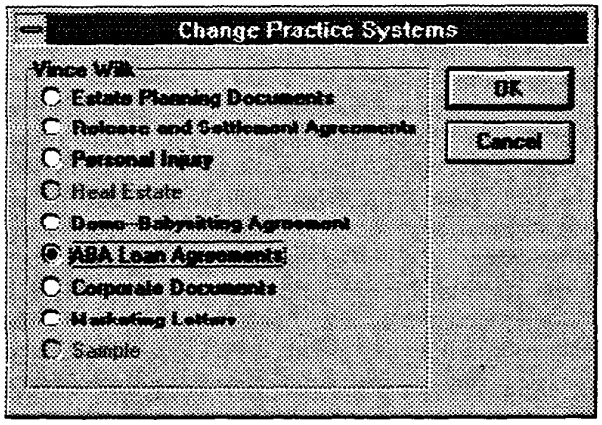


FIGURE 1: WinDraft's Practice Systems Dialog

What-you-see-is-what-you-get outline printing. An essential feature of WinDraft is its ability to print data in a format that is useful to lawyers and staff. Eidelman says the idea behind the data checklist stems from the old "Master Information List" (MIL) from the days of "manual substantive systems." (This feature is based on the ideas in the pioneering book written by incoming ABA President Roberta Cooper Ramo, How to Create a System for the Law Office.) The entire checklist, including headings, questions and answers, will be printed in outline form, with indenting and graying as on the screen. This printout provides an audit trail and is perfect for efficient review of the decisions that have been made by the document drafter.

Common interface. The program's menu choices and toolbars work in exactly the same way as the Microsoft Office programs. If you can use Windows, you can use WinDraft.

"Blasting Off" to Merge Documents

Once the data is set the way you want it, you click on the "Blast-off" button—a rocket icon—and WinDraft displays a "Blast-off dialog," which can be customized for each practice area or can be a simple "pick list" of the model documents. For example, at the ABA Document Assembly Shootout, it looked like the box in Figure 3.

WinDraft loads the model documents and merges them on the screen, displaying status messages and a progress bar. When the merge is completed, the user is left with a customized legal document that is ready to edit and print. If you use PC DOCS or another document management system, WinDraft will automatically save each new document, fill out each profile with custom information, and return a "hit list" of the new documents.

The Author's Perspective: Creating and Modifying Systems

Setting up and maintaining any practice system—even a manual one—can be a challenge. It can eat up lots of nonbillable time and requires teamwork. WinDraft is designed to provide an open, natural environment to support "programming" documents with legal rules. This is based on two key concepts:

1. WinDraft uses simple programming commands in documents.
2. Normal word processing documents are used for all model documents.

The WinDraft Language

WinDraft's simplicity extends to its programming commands. In practice, model documents are in a constant state of flux, changing as the law and needs of lawyers change. Also, a model document must be easily modifiable and "auditable" by lawyers who are not programmers.

Based on research by pioneer James Sprowl, WinDraft embeds legal rules and "programming" within brackets, stated in terminology that looks to a lawyer like "legalese." The system is "destructive" in that it scans a model document and deletes irrelevant text rather than building text out of a database of paragraphs. Systems composed of fragments or paragraphs are extremely difficult for a lawyer to audit.

Eidelman adopted this approach by employing a stripped-down programming language to embed legal rules in word processing documents. The basic logical command used is the "IF...ELSE...ENDIF" structure. As pointed out in the WinDraft manual, this is computer programming, but "It is what you learn in the first week of Programming 101."

Even though I had no programming background, I found that it was easy to work with IF statements, although this is not surprising since lawyers work with IF...THEN rules every day.

WinDraft also includes about 15 other commands you can use to include other model documents, set variables, display status information, and perform other functions.

A number of buttons on the WinDraft Author toolbar make it

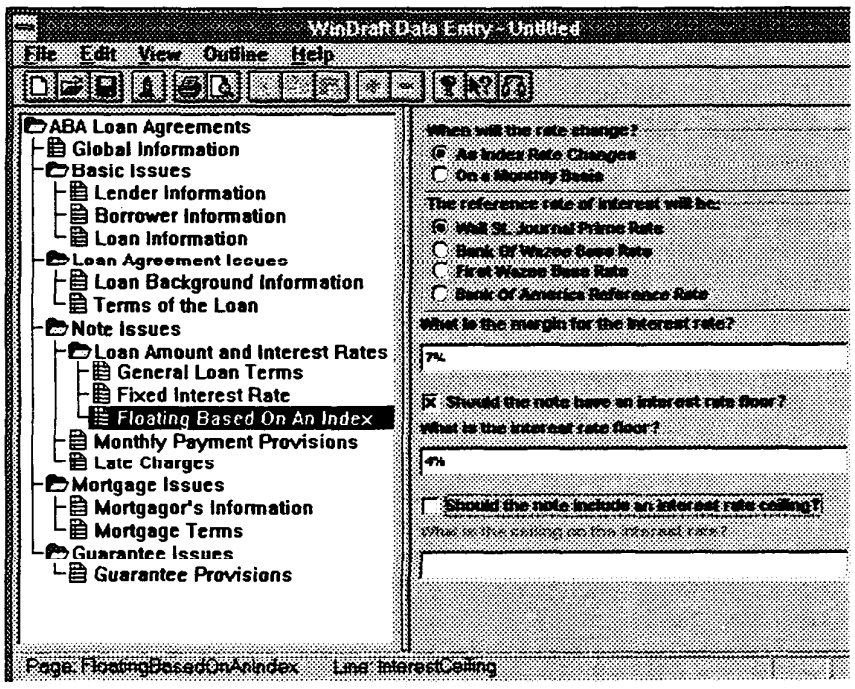


FIGURE 2: The WinDraft Data Entry Program

Turn to page 4

© ABA 1995 Reprinted with Permission

WinDraft 1.1

continued from page 3

easy to insert and format variables and commands.

Using Advanced Features of Word

The WinDraft manual recommends using the formatting and organizing features of the word processor as much as possible, and the samples do just that. The model documents are organized in Microsoft Word outlines, making it easier to work with complex documents. These outline headings are also used for formatting, paragraph numbering, cross-references and table of contents entries. No special programming is necessary. If they work in the model documents, they will work in the merged document.

I particularly like the clever way you can use Word fields to deal with changes of terminology in documents. For example, instead of inserting personal pronoun changes (his, her, their, its) as WinDraft variables, Eidelman recommends using Word fields for this. A special button on the WinDraft toolbar lets you turn the display of the pronoun fields on and off dynamically. This dramatically improves the readability of the model documents.

Observations

Unlike most of the comprehensive systems currently available, WinDraft is easy to learn. Like most lawyers, I lack a computer background. This was not a problem with WinDraft, however, because its features are intuitive and easy to master. In fact, WinDraft's greatest strength is its simplicity. Many programs are weighed down by features that only experts can appreciate or understand. WinDraft works with the word processor, not against it. Since everything is done through Microsoft Word, the systems are easy to maintain and update as the law or drafting conventions change. This will prevent the system from falling into disuse if the resident expert leaves the firm.

Another important WinDraft advantage is that model documents are easy to read and work with. In addition, WinDraft facilitates the team approach by allowing tasks to be assigned to an appropriate worker in the firm. For example, a lawyer works on the legal aspects of the system, while a secretary handles formatting issues. Because the parts of a system are all word processing documents, this is easy to do.

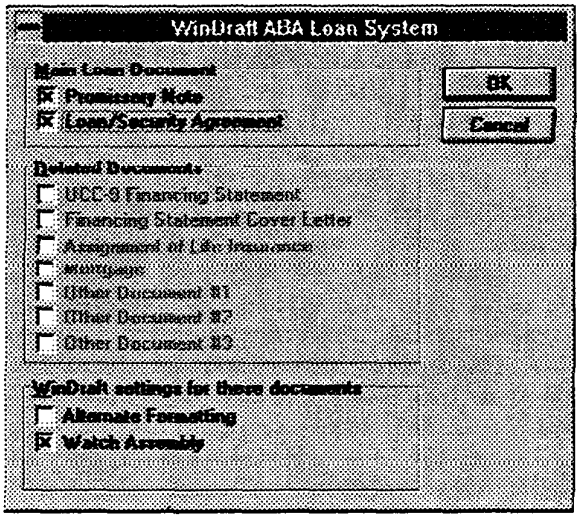


FIGURE 3

Creating the Client Checklist Scripts

The "scripts" that create the client data-entry checklists are edited in Word, or any other editor, and saved as an ASCII file by clicking on a button. There is a simple scripting language that lets you organize the buttons, checkboxes, text and headings in the outline. Data checklists are easy to build and modify on-the-fly.

Wish List

One problem with WinDraft's model documents is that extra hard returns can end up in the final document and be tricky to track down in the model document. It took some time to get the hang of where to put hard returns so that the formatting and spacing worked perfectly. Also, although you can imbed Excel spreadsheets into your documents and do math in Word tables, WinDraft currently lacks capability to handle math calculations within the WinDraft language. Eidelman Associates says this feature will be available soon.

Conclusion

Many programs ostensibly designed to save users time and money accomplish just the opposite, simply because they are too difficult to use. Such programs require constant support, resident experts and herculean patience from the user. This is a common problem with technological innovation—programs are written by techies for techies rather than with the average law firm's needs in mind. In creating WinDraft, Eidelman kept in mind the needs and limitations of lawyers and support staff members.

WinDraft deftly weaves clever features with a simple, commonsense interface, creating a program that is actually enjoyable to use. Users will find that WinDraft can readily adapt to changes in the law, the needs of a client or even a lawyer's practice. Firms and businesses interested in document assembly software should give WinDraft a close look.

- WinDraft , Eidelman Associates, 317 S. Division, Ste. 187, Ann Arbor, MI 48104; 800/77-LAPTOP; Internet: 73067.2131@compuserve.com. $499 single user license. Site licenses, network versions, add-on information is available from the vendor.

Vincent Wilk is a recent graduate of the University of Michigan Law School and will soon begin work as an associate with Nutter, McLennen and Fish in Boston.